THAILAND AUTOMOTIVE INSTITUTE
สถาบันยานยนต์

หลักสูตร

พื้นฐานระบบอิเล็กทรอนิกส์ในยานยนต์สมัยใหม่

วิชา ระบบการสื่อสาร

โครงการยกระดับผู้ผลิตชิ้นส่วนยานยนต์ไทย
เพื่อเตรียมความพร้อมในการเข้าสู่ห่วงโซ่อุปทานของ
อุตสาหกรรมยานยนต์สมัยใหม่
(Parts Transformation)

02-712-2414 www.thaiauto.or.th @thaiauto

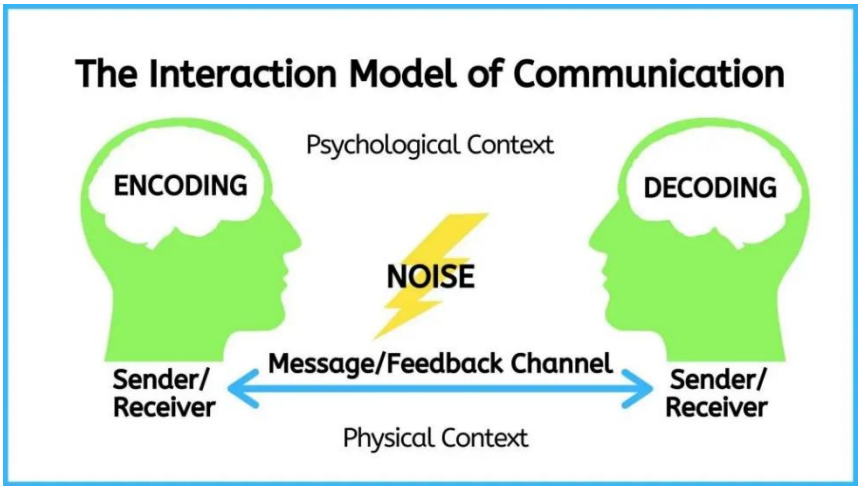
สารบัญ

สารบัญ (ต่อ)

บทที่ 1 พื้นฐานการสื่อสารข้อมูล

1.1 ความรู้เบื้องต้นเกี่ยวกับการสื่อสาร

การสื่อสารเป็นกระบวนการแลกเปลี่ยนข้อมูลระหว่างเครื่องส่ง (ผู้ส่ง) และเครื่องรับ (ผู้รับ) ตั้งแต่สองตัวขึ้นไปผ่านการใช้ช่องสัญญาณ โดยการสื่อสารข้อมูล จะเป็นการแลกเปลี่ยนข้อมูลระหว่างสองฝ่าย อุปกรณ์ต่างๆ ผ่านช่องทาง หรือการสื่อบางรูปแบบ เช่น สายไฟ (Wire) หรือจะเป็นสายอากาศ (Wireless) กล่าวคือสำหรับการเกิดการสื่อสารข้อมูล อุปกรณ์สื่อสาร จะต้องเป็นส่วนหนึ่งของระบบการสื่อสารที่ประกอบไปด้วย การรวมกันของอุปกรณ์ฮาร์ดแวร์หรือซอฟต์แวร์ และโปรแกรม โดยมีตัวอย่างโมเดลการสื่อสารดังรูปที่ 1



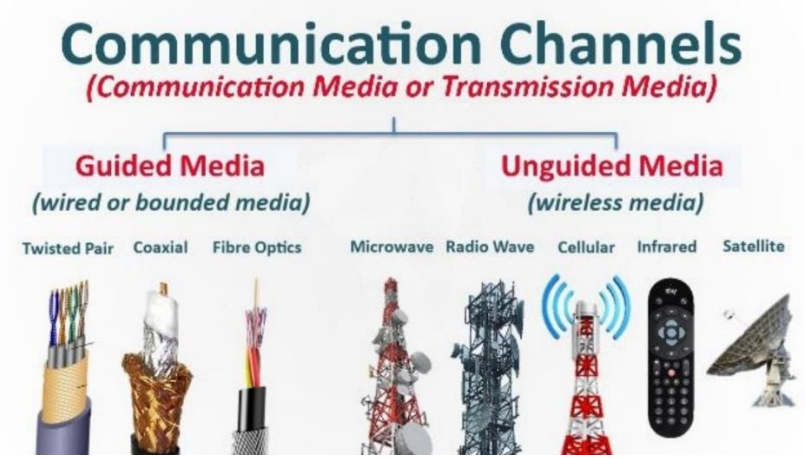
รูปที่ 1 โมเดลการสื่อสาร

ที่มา: https://botpenguin.com/glossary/communication-model

1.2 องค์ประกอบของการสื่อสารข้อมูล

ในระบบการสื่อสารข้อมูลนั้น ส่วนใหญ่แล้วจะมี 5 องค์ประกอบด้วยกัน ได้แก่

1. **ข้อมูล/ข้อความ (Data/Message)** โดยข้อมูลที่จะใช้ในการสื่อสารสามารถอยู่ในรูปแบบต่างๆได้ เช่น ข้อความ รูปภาพ เสียง หรือวิดีโอ
2. **เครื่องส่ง หรือผู้ส่ง (Transmitter/Sender)** เครื่องส่ง หรือผู้ส่งมักจะเป็นอุปกรณ์ ร่วมถึงระบบที่เป็นตัวเริ่มต้นกระบวนการสื่อสารโดยการเข้ารหัสในขั้นตอนแรก และนำส่งข้อมูล
3. **เครื่องรับ หรือผู้รับ (Receiver)** เครื่องรับ หรือผู้รับมักจะเป็นอุปกรณ์ ร่วมถึงระบบที่รับและถอดรหัสข้อมูลที่ส่งมา ทำให้ผู้รับสามารถเข้าใจการสื่อสารจากเครื่องส่ง หรือผู้ส่งได้
4. **ช่องสัญญาณ หรือสื่อกลาง (Transmission Channel/Medium)** ช่องสัญญาณ หรือสื่อกลางสามารถแบ่งได้เป็น ข้อมูลที่เดินทางผ่านทางระบบมีสาย (เช่น สายเคเบิล) และการสื่อสารไร้สาย (เช่น คลื่นวิทยุ) แสดงในรูปที่ 2



รูปที่ 2 รูปแบบช่องทางการสื่อสาร

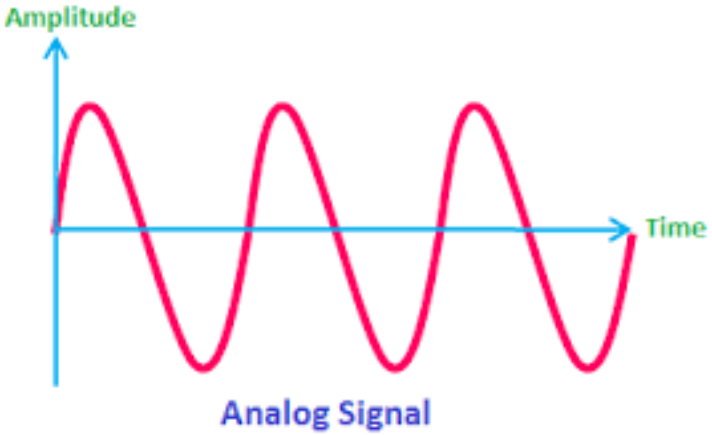
ที่มา: https://www.youtube.com/watch?app=desktop&v=doAVuehH-L4

5. **ชุดกฎเกณฑ์ (Set of Rules (Protocol)** ชุดกฎเกณฑ์ เป็นข้อตกลงที่ถูกกำหนดไว้ล่วงหน้า โดยมีมาตรฐานที่เป็นตัวคอยควบคุมรูปแบบการสื่อสาร และการจัดการข้อผิดพลาดในระหว่างกระบวนการสื่อสาร

1.3 การแสดงข้อมูลการสื่อสาร

การแสดงข้อมูลการสื่อสาร คือวิธีการหรือเทคนิคที่ใช้ เพื่อเข้าถึงรหัสข้อมูล ในรูปแบบที่เข้าใจได้ และสามารถประมวลผลได้ด้วยระบบคอมพิวเตอร์ในการคำนวณ โดยข้อมูลสามารถมีได้หลายรูปแบบสัญญาณ คือ

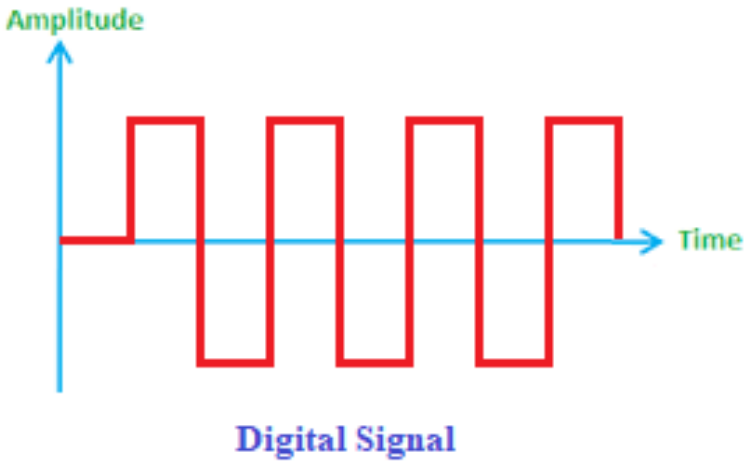
สัญญาณอนาล็อก (Analog Signal) หมายถึง สัญญาณข้อมูลแบบต่อเนื่อง (Continuous Data) มีขนาดของสัญญาณไม่คงที่ มีการเปลี่ยนแปลงขนาดของสัญญาณแบบค่อยเป็นค่อยไป มีลักษณะเป็นเส้นโค้งต่อเนื่องกันไป โดยการส่งสัญญาณแบบอนาล็อกจะถูกรบกวนให้มีการแปลความหมายผิดพลาดได้ง่าย เช่น สัญญาณเสียงในสายโทรศัพท์ เป็นต้น โดยตัวอย่างสัญญาณอนาล็อก เป็นดังรูปที่ 3



รูปที่ 3 สัญญาณ analog

ที่มา: https://www.primusthai.com/primus/Knowledge/info?ID=132

สัญญาณดิจิตอล (Digital Signal) หมายถึง สัญญาณที่เกี่ยวข้องกับข้อมูลแบบไม่ต่อเนื่อง (Discrete Data) ที่มีขนาดแน่นอนซึ่งขนาดดังกล่าวอาจกระโดดไปมาระหว่างค่าสองค่า คือ สัญญาณระดับสูงสุดและสัญญาณระดับต่ำสุด ซึ่งสัญญาณดิจิตอลนี้เป็นสัญญาณที่คอมพิวเตอร์ใช้ในการทำงานและติดต่อสื่อสารกันเป็นค่าของเลขลงตัว โดยปกติมักแทนด้วยระดับแรงดันที่แสดงสถานะเป็น 0 และ 1 หรืออาจจะมีหลายสถานะซึ่งจะกล่าวถึงในเรื่องระบบสื่อสารดิจิตอลมีค่าที่ตั้งไว้ เป็นค่าบอกสถานะ ถ้าสูงเกินค่าที่ตั้งไว้สถานะเป็น 1 ถ้าต่ำกว่าค่าที่ตั้งไว้สถานะเป็น 0 ซึ่งมีข้อดีในการทำให้เกิดความผิดพลาดน้อยลง โดยตัวอย่างสัญญาณดิจิตอล เป็นดังรูปที่ 4

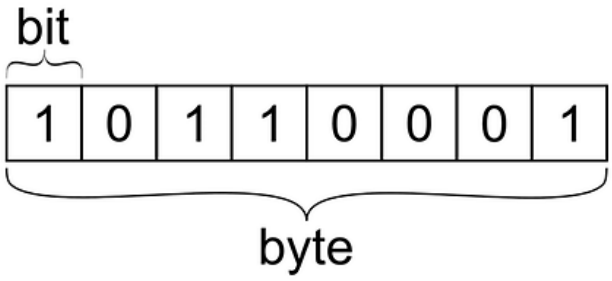


รูปที่ 4 สัญญาณ Digital

ที่มา: https://www.primusthai.com/primus/Knowledge/info?ID=132

1.4 แผนการเข้าถึงรหัสข้อมูล

แผนการเข้าถึงรหัสข้อมูลนั้น เกี่ยวข้องกับการแปลงข้อมูล เป็นรูปแบบดิจิทัลที่สามารถทำให้อุปกรณ์อ่านได้ โดยใช้การเข้ารหัสรูปแบบที่เป็นส่วนสำคัญของข้อมูลในลำดับชั้นของข้อมูลที่เล็กที่สุด(Bit) และหน่วยของข้อมูลที่เป็นเลขฐานสองจำนวน 8 หลัก (Byte) โดยตัวอย่างการเปรียบเทียบ ของ Bit และ Byte แสดงในรูปที่ 5



รูปที่ 5 Bit และ Byte
ที่มา: https://noomerzx.medium.com/bit-byte-%E0%B9%81%E0%B8%A5%E0%B8%B0-characters-set-1d056994b69b

1.4.1 บิต (Bit)

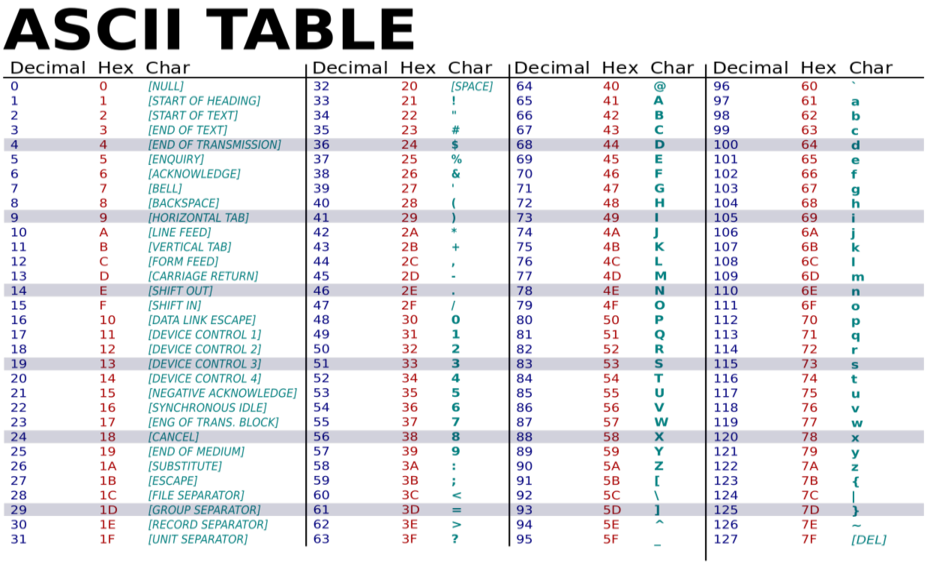
Bit ย่อมาจาก binary digit คือลำดับชั้นของข้อมูลที่เล็กที่สุด ดังที่ทราบกันดีว่าข้อมูลที่จะทำงานร่วมกับคอมพิวเตอร์ได้นั้น จะต้องเอามาแปลงให้อยู่ในรูปของเลขฐานสองเสียก่อนคอมพิวเตอร์ถึงจะเข้าใจ และทำงานตามที่ต้องการ เมื่อแปลงแล้วจะได้ตัวเลขแทนสถานะเปิดและปิดของสัญญาณไฟฟ้าที่เรียกว่า Bit เพียงสองค่านั้น คือ Bit 0 และ Bit 1 ส่วนเลขฐาน 2 นั้นถูกนำมาใช้ในทางคอมพิวเตอร์ เพราะว่าเลข 0 กับเลข 1 สามารถแทนสถานการณ์ 2 อย่างคือ ปิดและเปิด หรือ ไม่จริงกับจริง ซึ่งสามารถนำไปใช้ได้กับระดับแรงดันไฟฟ้าในวงจรของเครื่องมืออีเล็กทรอนิกส์ได้พอดี ระบบเลขฐาน 2 มีความสำคัญมากในการคำนวณแบบดิจิตอล

1.4.2 ไบต์ (Byte)

ไบต์ (Byte) หมายถึง หน่วยของข้อมูลที่เป็นเลขฐานสองจำนวน 8 หลัก หรือ 8 บิต ที่ใช้แทนข้อมูลที่เป็นตัวอักษร ตัวเลข หรือสัญลักษณ์ต่าง ๆ เพียง 1 ตัว ตามรหัสแอสกี (ASCII) เช่น A B C ก ข ค ง เป็นต้น หรือจำนวนเต็ม 1 จำนวน เช่น 01000001 คือ ตัว A หรือ 01100010 คือ ตัว B โดย 8 บิตเท่ากับ 1 ไบต์ ไบต์จึงเป็นหน่วยข้อมูลที่มีขนาดใหญ่กว่าบิทและนิยมใช้เป็นหน่วยวัดความจุในการเก็บข้อมูลในคอมพิวเตอร์หรือสื่อบันทึกข้อมูลด้วย

1.4.3 ASCII

ย่อมาจาก American Standard Code for Information Interchange (ASCII) กล่าวคือเป็นการกำหนดมาตรฐานการแปลงไปแปลงมาระหว่าง เลขฐาน16 (HEX code) เป็น ตัวอักษรภาษาอังกฤษ โดยมาตาราง ASCII ดังรูปที่ 6



ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

รูปที่ 6 ตาราง ASCII

ที่มา: https://payat-jira.medium.com/ascii-code

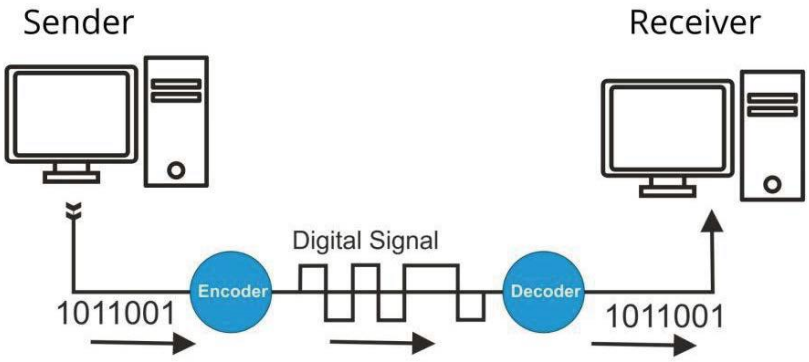
1.5 การส่งข้อมูล (Data transmission)

การส่งข้อมูลคือกระบวนการส่งข้อมูลดิจิทัลหรืออนาล็อกผ่านสื่อการสื่อสารไปยังคอมพิวเตอร์เครือข่ายการสื่อสาร หรืออุปกรณ์อิเล็กทรอนิกส์ตั้งแต่หนึ่งเครื่องขึ้นไป ช่วยให้สามารถถ่ายโอนและสื่อสารอุปกรณ์ในสภาพแวดล้อมแบบจุดต่อจุด จุดต่อหลายจุด และหลายจุดต่อหลายจุดได้

1.5.1 การส่งผ่านข้อมูลแบบดิจิตอล (Digital Transmission)

การส่งผ่านข้อมูลแบบดิจิตอล คือ กระบวนการที่ให้ข้อมูลส่งออกจากผู้ส่ง ผ่านตัวกลางหรือสายสัญญาณ ไปยังผู้รับอย่างถูกต้อง แสดงในรูปที่ 7 โดยจำเป็นต้องผ่านการดำเนินการตามขั้นตอน มีดังนี้ การแปลงข้อมูลให้เป็นสัญญาณ การส่งสัญญาณข้อมูลผ่านตัวกลาง และกลับเป็นข้อมูลตามเดิม เป็นกระบวนการที่ให้ข้อมูลส่งออกจากผู้ส่ง ผ่านตัวกลางหรือสายสัญญาณ ไปยังผู้รับอย่างงถูกต้อง โดยจำเป็นต้องผ่านการดำเนินการตามขั้นตอน ดังนี้

1. การแปลงข้อมูลให้เป็นสัญญาณ
2. การส่งสัญญาณข้อมูลผ่านตัวกลาง
3. การแปลงสัญญาณข้อมูลกลับเป็นข้อมูลตามเดิม

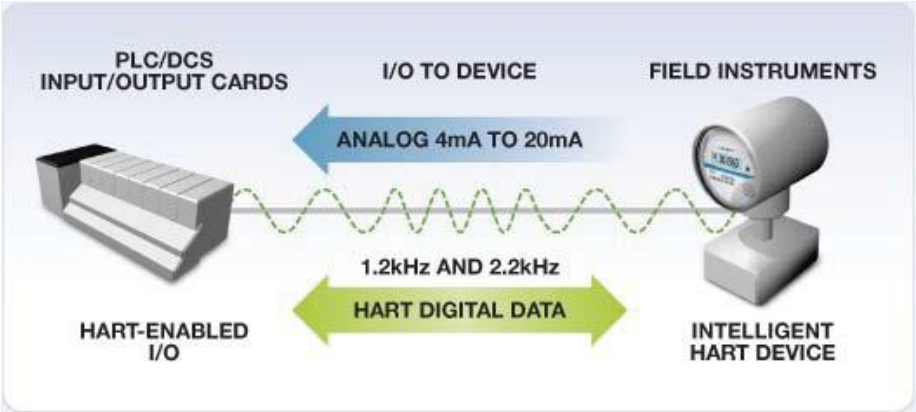


รูปที่ 7 การส่งข้อมูลแบบ Digital

ที่มา: https://www.what-when-how.com/data-communication/digital-transmission/

1.5.2 การส่งผ่านข้อมูลแบบอนาล็อก (Analog Transmission)

การส่งผ่านข้อมูลแบบอนาล็อก นั้นเป็นวิธีการส่งข้อมูลโดยการใช้สัญญาณต่อเนื่อง ซึ่งแปรผันตามค่าแอมพลิจูด เฟส หรือคุณสมบัติอื่นๆ ตามสัดส่วนของข้อมูลนั้นๆ แสดงในรูปที่ 8

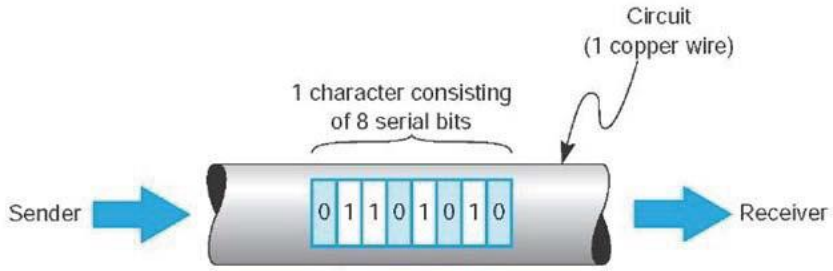


รูปที่ 8 การส่งข้อมูลแบบ Analog

ที่มา: https://www.analog.com/en/resourse/articles/hart-communication-network.html

1.5.3 การส่งผ่านข้อมูลแบบ ASCII (ASCII Data Transmission)

ข้อมูล ASCII นั้นจะถูกส่งผ่าน วงจรการสื่อสารตามลำดับทีละนิดและข้อมูลทั้งหมดจะต้องถูกส่งผ่านตัวกลางที่เพียงสิ่งเดียวคือสายไฟ รวมถึงอุปกรณ์การส่งสัญญาณจะค่อยๆส่งข้อมูลที่ละนิด 1 bit ไป 2 bit และส่งไปจนกว่าบิตทั้งหมดจะถูกส่ง โดยจะทำการวนส่ง bit ซ้ำๆ โดยแสดงตัวอย่างในรูปที่ 9



รูปที่ 9 การส่งข้อมูลแบบ ASCII

ที่มา: https://www.what-when-how.com/data-communication/digital-transmission/

1.6 ทิศทางการส่งข้อมูล (Transmission Mode)

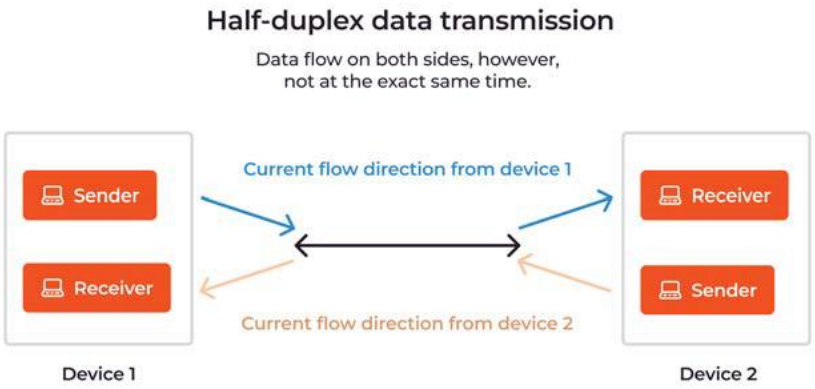
การส่งข้อมูลหรือที่เรียกว่าโหมดการสื่อสารเป็นวิธีการถ่ายโอนข้อมูลระหว่างอุปกรณ์บนบัสและเครือข่ายที่ออกแบบมาเพื่ออำนวยความสะดวกในการสื่อสาร แบ่งออกเป็นสามประเภท

- **Simplex** ในโหมด Simplex การสื่อสารจะมีทิศทางเดียวเช่นเดียวกับบนถนนเดินรถทางเดียว มีเพียงหนึ่งในสองอุปกรณ์บนลิงค์เท่านั้นที่สามารถส่งได้ ส่วนอีกเครื่องสามารถรับได้เท่านั้น โหมดซิมเพล็กซ์สามารถใช้ความจุทั้งหมดของช่องสัญญาณเพื่อส่งข้อมูลไปในทิศทางเดียว ตัวอย่างเช่น คีย์บอร์ดและจอภาพแบบดั้งเดิม แป้นพิมพ์สามารถป้อนได้เฉพาะอินพุต ส่วนจอภาพสามารถให้เอาต์พุตได้เท่านั้น ตัวอย่างการส่งข้อมูล Simplex แสดงในรูปที่ 10

รูปที่ 10 การส่งข้อมูลแบบ Simplex

ที่มา: https://gcore.com/learning/data-transmission-guide-everthing-you-need-to-know/

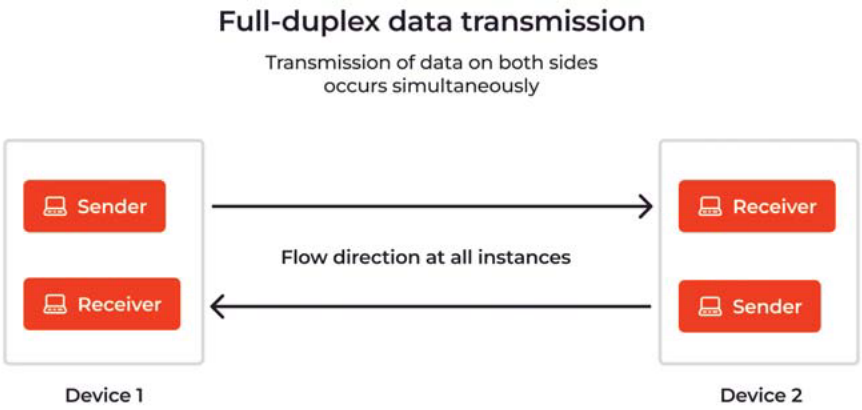
- **Half-Duplex** ในโหมดฮาล์ฟดูเพล็กซ์แต่ละสถานีสามารถส่งและรับได้ แต่ไม่ใช่ในเวลาเดียวกัน เมื่ออุปกรณ์เครื่องหนึ่งกำลังส่ง อุปกรณ์อีกเครื่องจะรับได้เท่านั้น และในทางกลับกัน โหมดฮาล์ฟดูเพล็กซ์ใช้ในกรณีที่ไม่จำเป็นต้องสื่อสารทั้งสองทิศทางในเวลาเดียวกัน สามารถใช้ความจุทั้งหมดของช่องได้ในแต่ละทิศทาง ตัวอย่างเช่น เครื่องส่งรับวิทยุซึ่งส่งข้อความครั้งละหนึ่งข้อความและส่งข้อความทั้งสองทิศทางตัวอย่างการส่งข้อมูล Half-Duplex แสดงในรูปที่ 11



รูปที่ 11 การส่งข้อมูลแบบ Half-Duplex

ที่มา: https://gcore.com/learning/data-transmission-guide-everthing-you-need-to-know/

- **Full-Duplex** ในโหมดฟูลดูเพล็กซ์ ทั้งสองสถานีสามารถส่งและรับพร้อมกันได้ โดยสัญญาณที่ไปในทิศทางเดียวจะแชร์ความจุของลิงก์กับสัญญาณที่ไปในทิศทางอื่น การแบ่งปันนี้สามารถเกิดขึ้นได้สองวิธี: คือ 1. ลิงก์จะต้องมีเส้นทางการรับส่งข้อมูลสองเส้นทางแยกจากกัน เส้นทางหนึ่งสำหรับการส่งและอีกเส้นทางหนึ่งสำหรับการรับ 2. ความจุจะถูกแบ่งระหว่างสัญญาณที่เคลื่อนที่ทั้งสองทิศทาง โดยในโหมดฟูลดูเพล็กซ์จะใช้เมื่อต้องมีการสื่อสารทั้งสองทิศทางตลอดเวลา อย่างไรก็ตามความจุของช่องจะต้องแบ่งระหว่างสองทิศทาง ตัวอย่าง เครือข่ายโทรศัพท์ซึ่งมีการสื่อสารระหว่างบุคคลสองคนด้วยสายโทรศัพท์ ซึ่งทั้งสองสามารถพูดคุยและฟังพร้อมกันได้ ตัวอย่างการส่งข้อมูล Full-Duplex แสดงในรูปที่ 12

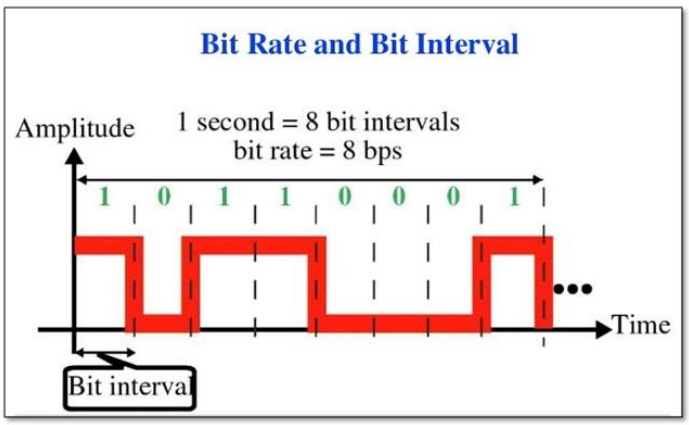


รูปที่ 12 การส่งข้อมูลแบบ Full-Duplex

ที่มา: https://gcore.com/learning/data-transmission-guide-everthing-you-need-to-know/

1.7 อัตราการส่งข้อมูล (Transmission rate)

คือความเร็วที่ข้อมูลใดที่ถูกถ่ายโอนจากอุปกรณ์หรือวงจร โดยมักจะวัดเป็นหน่วยที่เกี่ยวข้องกับจำนวนข้อมูลที่ถ่ายโอน ต่อรอบ เช่น ตัวอักษรต่อวินาที หรือบิตต่อวินาที (bps) แสดงในรูปที่ 13



รูปที่ 13 อัตราการส่ง Bit

ที่มา: https://kunaldube.blogspot.com/2017/04/bit-rate-baud-rate-bit-length-bit.html

1.8 รูปแบบในการส่งข้อมูล (Data Transmission)

การส่งข้อมูลในระบบเครือข่าย สามารถทำได้ 2 ลักษณะ คือ การส่งแบบขนาน และการส่งแบบอนุกรม

1.8.1 การส่งแบบขนาน (Parallel transmission)

คือการส่งข้อมูลพร้อมกันทีละหลาย ๆ บิตในหนึ่งรอบสัญญาณนาฬิกา โดยการส่งจะรวมบิต 0 และ 1 หลาย ๆ บิตเข้าเป็นกลุ่มจำนวน n บิต ผู้ส่งส่งครั้งละ n บิต ผู้รับจะรับครั้งละ n บิตเช่นกัน ซึ่งจะคล้ายกับการพูดคุยโดยจะพูดเป็นคำ ๆ ไม่พูดทีละตัวอักษร กลไกการส่งข้อมูลแบบขนานใช้หลักการง่าย ๆ เมื่อส่งครั้งละ n บิต ต้องใช้สาย n เส้น แต่ละบิตมีสายของตนเอง ในการส่งแต่ละครั้งทุกเส้นต้องใช้สัญญาณเวลาเดียวกัน ทำให้สามารถส่งออกไปยังอุปกรณ์อื่นพร้อมกันได้

1.8.2 การส่งข้อมูลแบบอนุกรม (Serial transmission)

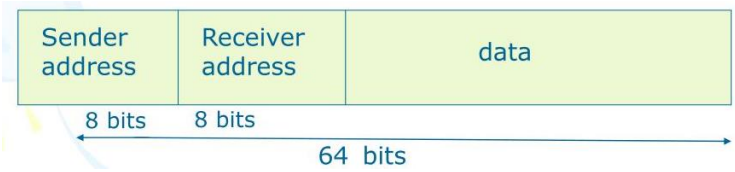
การส่งข้อมูลแบบอนุกรม จะใช้วิธีการส่งทีละ 1 บิตในหนึ่งรอบสัญญาณเวลา ทำให้ดูเหมือนว่าบิตต่าง ๆ เรียงต่อเนื่องกันไป จากอุปกรณ์หนึ่งไปยังอีกอุปกรณ์หนึ่ง กล่าวคือ การใช้ช่องทางการสื่อสารเพียง 1 ช่อง ช่วยทำให้ลดค่าใช้จ่ายลง แต่ข้อเสียคือ ความเร็วของการส่งที่ต่ำ ตัวอย่างของการส่งข้อมูลแบบอนุกรม เช่น โมเด็มจะใช้การส่งแบบอนุกรมเนื่องจากในสัญญาณโทรศัพท์มีสายสัญญาณเส้นเดียว และอีกเส้นหนึ่งเป็นสายดิน

บทที่ 2 พื้นฐานการสื่อสารโปรโตคอล และเครือข่าย

2.1 โปรโตคอลการสื่อสาร (Communication Protocols)

โปรโตคอลการสื่อสารเป็นชุดของกฎเกณฑ์ที่กำหนดวิธีการสื่อสารระหว่างเอนทิตีในระบบการสื่อสารที่มีมากกว่าหนึ่งหน่วยงาน โดยโปรโตคอลนี้จะครอบคลุมการส่งข้อมูลทั้งในแง่ของรูปแบบ (ไวยากรณ์) และความหมายของข้อมูล (ความหมาย) รวมถึงการประสานเวลา (ซิงโครไนซ์) เพื่อให้การสื่อสารเป็นไปอย่างมีประสิทธิภาพ นอกจากนี้ยังมีการจัดการข้อผิดพลาดเพื่อให้สามารถกู้คืนข้อมูลได้หากเกิดปัญหาในการสื่อสาร โดยองค์ประกอบสำคัญของการสื่อสารโปรโตคอล มี 3 สิ่ง คือ

1 **ไวยากรณ์ (Syntax)** หมายถึงรูปแบบของข้อมูล ซึ่งบ่งบอกลำดับการนำเสนอข้อมูล และวิธีการอ่านข้อมูลนั้น ๆ ซึ่งเป็นวิธีการที่ใช้ในการแทนข้อมูล ตัวอย่างเช่น หากเราสมมติว่าชุดข้อมูล (data packet) มี 16 บิต โดย 4 บิตแรกเป็นที่อยู่ของผู้ส่ง (sender's address) 4 บิตสุดท้ายเป็นที่อยู่ของผู้รับ (receiver's address) และที่เหลือคือข้อความ (message) ดังนั้น นี่คือไวยากรณ์ที่ใช้ในการแทนบิตของข้อมูล แสดงตัวอย่างในรูปที่ 14



รูปที่ 14 Syntax

ที่มา: http://polprog.net/blog/serial/

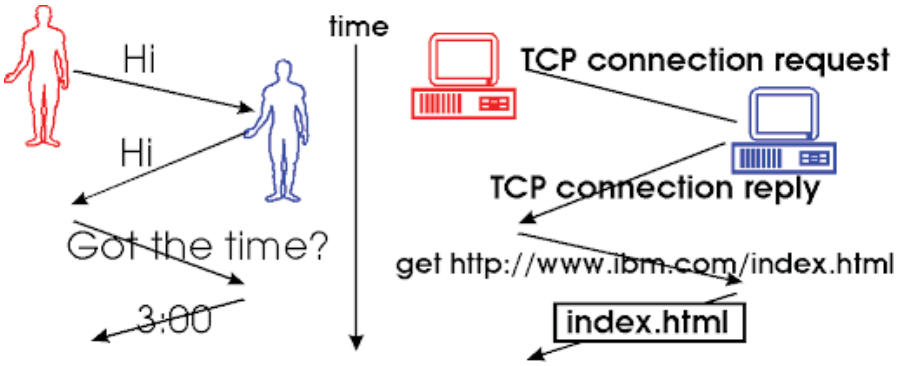
2 **ความหมาย (Semantics)** คือความหมายของแต่ละส่วนที่กล่าวถึงในไวยากรณ์ (syntax) ซึ่งรวมถึงข้อมูลการควบคุมสำหรับการประสานงานและการจัดการข้อผิดพลาด นอกจากนี้ยังระบุว่าไฟล์ใดกำหนดการกระทำใดบ้าง แสดงตัวอย่างในรูปที่ 15

St	Start bit, always low
(n)	Data bits (0 to 8)
P	Parity bit. Can be odd or even
Sp	Stop bit, always high
IDLE	No transfers on the communication line (RxD or TxD). An IDLE line must be high

รูปที่ 15 Semantics

ที่มา: http://polprog.net/blog/serial/

3 **การซิงโครไนซ์ (Synchronize)** การซิงโครไนซ์หมายถึงเวลาที่ข้อมูลจะถูกส่งและความเร็วที่ข้อมูลสามารถส่งได้ ตัวอย่างเช่น หากผู้ส่งส่งข้อมูลที่ความเร็ว 100 Mbps และผู้รับรับข้อมูลที่ความเร็ว 1 Mbps ข้อมูลจะล้นที่ปลายทางของผู้รับ แสดงตัวอย่างในรูปที่ 16

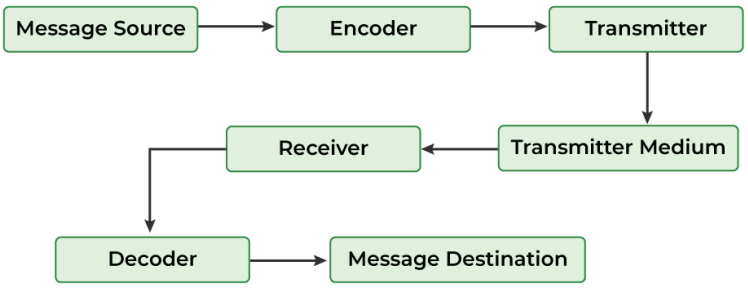


รูปที่ 16 การซิงโครไนซ์

ที่มา: http://www2.ic.uff.br/michael/kr1999/1-introduction/102-protocol.htm

นอกจากนี้แล้วโปรโตคอลการสื่อสารเครือข่ายยังต้องการองค์ประกอบดังต่อไปนี้

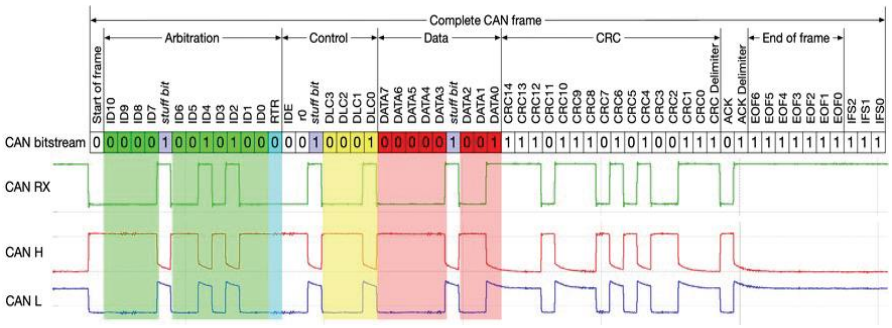
1. **การเข้ารหัสข้อความ (Message Encoding)** ข้อความจากผู้ส่งจะถูกเข้ารหัสเป็นสัญญาณหรือคลื่น จากนั้นส่งผ่านสื่อที่มีสายหรือไร้สายไปยังปลายทาง แสดงดังรูปที่ 17 และทำการถอดรหัสก่อนส่งข้อความถึงปลายทาง การเข้ารหัสเป็นกระบวนการที่เปลี่ยนชุดตัวอักษรยูนิโค้ด (Unicode) เป็นลำดับของไบต์



รูปที่ 17 Message Encoding

ที่มา: https://www.geeksforgeeks.org/elements-of-network-protocol/

2. **การจัดรูปแบบและการหุ้มข้อมูล (Message Formatting and Encapsulation)**: มีรูปแบบที่ตกลงกันระหว่างผู้ส่งและผู้รับ ซึ่งใช้ในการหาข้อมูลเพื่อระบุผู้ส่งและผู้รับอย่างถูกต้อง รูปแบบของข้อความจะขึ้นอยู่กับประเภทของข้อความและสื่อที่ใช้ในการส่ง ข้อความจะถูกหุ้มเพื่อใส่ข้อความหนึ่งเข้าไปในข้อความอื่นเพื่อการส่งจากแหล่งที่มาไปยังปลายทาง โดยยกตัวอย่าง เครื่องหมาย Start-Of-Frame (SOF) และ End-Of-Frame (EOF) ถูกใช้สำหรับการหุ้มข้อมูล (encapsulation) แสดงในรูปที่ 18



รูปที่ 18 CAN frame

ที่มา: http://en.wikipedia.org/wiki/CANBUS

3. **ขนาดข้อความ (Message Size)**: ข้อความยาวต้องแบ่งออกเป็นชิ้นเล็กๆ เพื่อส่งผ่านเครือข่าย หรือกระบวนการแบ่งข้อความยาวเป็นชิ้นๆ ก่อนการส่งผ่านเครือข่าย ตัวอย่างเช่น ในโทรศัพท์มือถือ SMS จำกัดขนาดข้อความไว้ที่ 160 ตัวอักษรปกติ แต่สำหรับตัวอักษรที่ไม่ใช่ตัวอักษรปกติจะต้องใช้ข้อมูล 16 บิตในการแทนที่ ส่งผลให้ขนาดข้อความถูกจำกัดไว้ที่ 70 ตัวอักษร
4. **การจัดการเวลา (Message Timing)**: การจัดการการไหลของข้อมูล เช่น การตอบกลับและการหมดเวลา (timeout) ต้องการข้อมูลควบคุมเวลาเฉพาะ การตรวจสอบความล่าช้าในการส่งข้อมูล และรวมถึงกฎเช่น วิธีการเข้าถึง, การควบคุมการไหล, และการหมดเวลาในการตอบกลับ

2.1 รูปแบบโครงข่าย (Network Topology)

รูปแบบโครงข่าย คือ จัดเรียงขององค์ประกอบต่างๆ เช่น โหนด (nodes), ลิงก์ (links), และอุปกรณ์ (devices) ในเครือข่ายคอมพิวเตอร์ ซึ่งกำหนดว่ามีการเชื่อมต่อและปฏิสัมพันธ์ระหว่างองค์ประกอบเหล่านี้อย่างไร การเข้าใจประเภทต่างๆ ของรูปแบบโครงข่ายช่วยในการออกแบบเครือข่ายที่มีประสิทธิภาพและมีความทนทาน รูปแบบทั่วไปได้แก่ Point to point, Bus

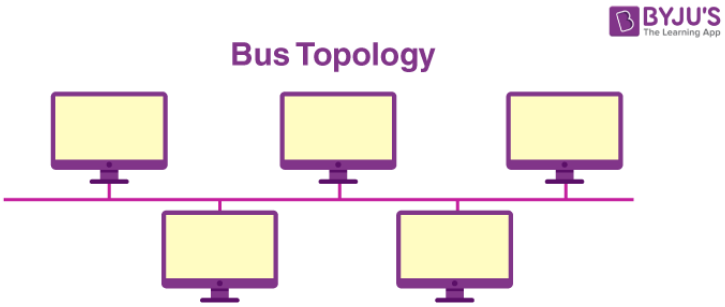
2.1.1 รูปแบบโครงข่ายแบบจุดต่อจุด (Point-to-Point)

รูปแบบโครงข่ายแบบจุดต่อจุดเป็นประเภทของรูปแบบโครงข่ายที่ทำงานตามฟังก์ชันของผู้ส่งและผู้รับ มันเป็นการสื่อสารที่เรียบง่ายที่สุดระหว่างสองโหนด โดยที่หนึ่งโหนดทำหน้าที่เป็นผู้ส่ง (sender) และอีกโหนดหนึ่งทำหน้าที่เป็นผู้รับ (receiver) แสดงตัวอย่างในรูปที่ 19

รูปที่ 19 Point to point Topology ที่มา: https://www.geeksforgeeks.org/difference-between-point-to-point-link-and-star-topology-network/

2.1.2 รูปแบบโครงข่ายแบบบัส (Bus)

รูปแบบโครงข่ายแบบบัสคือการตั้งค่าเครือข่ายที่อุปกรณ์ทั้งหมดเชื่อมต่อผ่านสายเคเบิลเดียวซึ่งเรียกว่า "บัส" บัสทำหน้าที่เป็นสื่อกลางในการสื่อสารร่วมกัน ทำให้ทุกอุปกรณ์ในเครือข่ายสามารถรับสัญญาณเดียวกันได้พร้อมกัน แสดงตัวอย่างในรูปที่ 20 โดยรูปแบบโครงข่ายแบบบัสมีความเรียบง่ายและเชื่อถือได้ เหมาะสำหรับเครือข่ายขนาดเล็ก ใช้สายเคเบิลน้อยจึงมีค่าใช้จ่ายต่ำ และสามารถขยายระบบได้ง่ายโดยการเชื่อมสายเคเบิลหลายเส้นเข้าด้วยกันเพื่อเพิ่มจำนวนคอมพิวเตอร์ในเครือข่าย

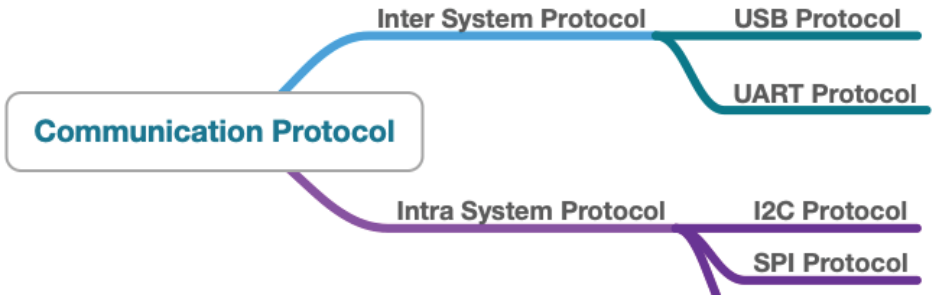


รูปที่ 20 Bus Topology

ที่มา: https://byjus.com/gate/bus-topology-notes/

2.2 ประเภทของโปรโตคอลการสื่อสาร (Types of Communication Protocols)

โปรโตคอลการสื่อสารทำหน้าที่เป็นหลักการในการแลกเปลี่ยนข้อมูลระหว่างระบบต่างๆ และภายในส่วนประกอบที่ซับซ้อนของระบบเดียวกัน มีโปรโตคอลการสื่อสารสองประเภทที่จำแนกได้ดังรูปที่ 21



รูปที่ 21 ประเภทของการสื่อสาร

ที่มา: http://fr.farnell.com/communication-network-protocol-trc-ar

2.2.1 โปรโตคอลระหว่างระบบ (Inter System Protocol)

เป็นโปรโตคอลที่ช่วยในการสร้างการสื่อสารระหว่างอุปกรณ์สองตัว ตัวอย่างที่ดีคือการเชื่อมต่อระหว่างคอมพิวเตอร์กับบอร์ดพัฒนา (development board) ผ่านระบบบัสระหว่างกัน (inter bus system) โดยโปรโตคอลระหว่างระบบมี 2 ประเภทคือ

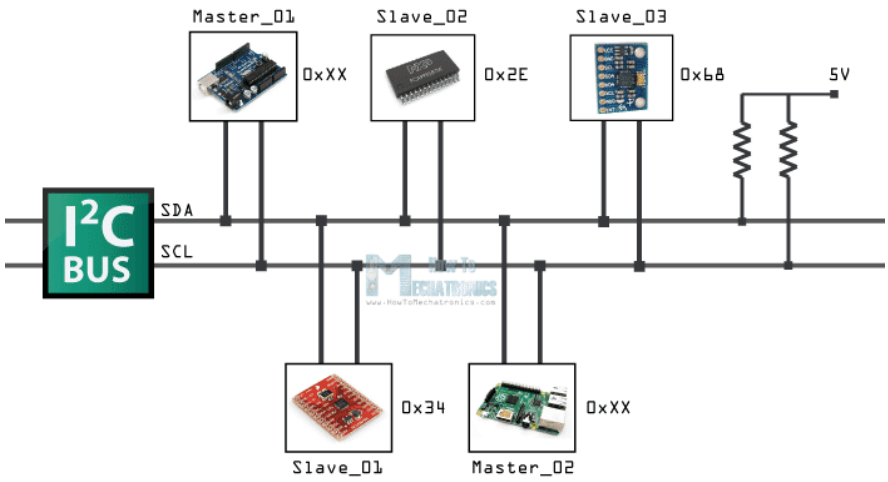
1 **โปรโตคอลการสื่อสาร USB (Universal Serial Bus)** เป็นโปรโตคอลการสื่อสารแบบอะซิงโครนัสที่รองรับการแลกเปลี่ยนข้อมูลและการถ่ายโอนพลังงานระหว่างอุปกรณ์อิเล็กทรอนิกส์ต่างๆ โดยเป็นมาตรฐานอุตสาหกรรมที่ใช้เชื่อมต่ออุปกรณ์เสริมกับตัวควบคุมหลัก (host controller)

2 **โปรโตคอลการสื่อสาร UART (Universal Asynchronous Receiver/Transmitter)** เป็นโปรโตคอลซีเรียลที่ง่าย ซึ่งช่วยให้การถ่ายโอนข้อมูลระหว่างอุปกรณ์สองตัวทำได้ทีละบิต ตามความหมายแล้ว UART เป็นโปรโตคอลการสื่อสารฮาร์ดแวร์ที่ใช้การสื่อสารแบบซีเรียลอะซิงโครนัส (asynchronous serial communication) โดยสามารถปรับความเร็วได้ อะซิงโครนัสหมายความว่าไม่มีสัญญาณนาฬิกา (clock signal) เพื่อทำการซิงโครไนซ์บิตที่ออกจากอุปกรณ์ส่งไปยังปลายทางที่รับข้อมูล

2.2.2 โปรโตคอลภายในระบบ (Intra System Protocol)

โปรโตคอลภายในระบบสร้างการสื่อสารระหว่างส่วนประกอบต่างๆ ภายในแผงวงจร ในระบบฝังตัว (embedded systems) โปรโตคอลภายในระบบช่วยเพิ่มจำนวนของส่วนประกอบที่เชื่อมต่อกับตัวควบคุม การเพิ่มจำนวนส่วนประกอบนำไปสู่ความซับซ้อนของวงจรและการใช้พลังงานที่เพิ่มขึ้น โปรโตคอลภายในระบบยังรับประกันการเข้าถึงข้อมูลจากอุปกรณ์เสริมอย่างปลอดภัย โดยโปรโตคอลภายในระบบมี 3 ประเภทคือ

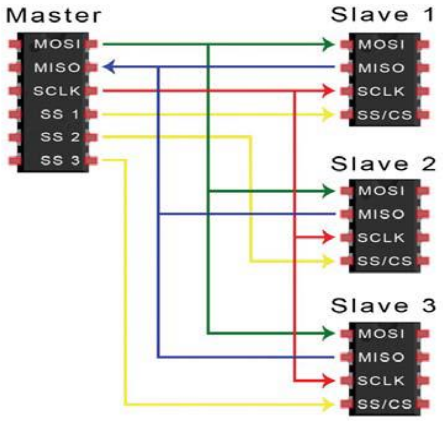
1. **โปรโตคอลการสื่อสาร I2C** การสื่อสาร I2C เป็นที่นิยมอย่างมากและใช้กันอย่างแพร่หลายโดยอุปกรณ์อิเล็กทรอนิกส์หลายชนิด เนื่องจากสามารถนำไปใช้ในแบบจำลองอิเล็กทรอนิกส์ได้ง่าย ซึ่งต้องการการสื่อสารระหว่างอุปกรณ์หลัก (master) กับอุปกรณ์ทาส (slave) หลายตัว หรือแม้กระทั่งระหว่างอุปกรณ์หลักหลายตัว ตัวอย่างในรูปที่ 22



รูปที่ 22 โปรโตคอล I2C

ที่มา: http://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/

2. **โปรโตคอลการสื่อสาร SPI** เป็นบัสข้อมูลแบบซีเรียลที่ใช้สาย 4 เส้น โดยมีโหนดเดียวที่เรียกว่า master ซึ่งจัดการกิจกรรมบนบัส ส่วนโหนดอื่นๆ ที่เรียกว่า slaves จะตอบสนองต่อ master สายทั้ง 4 เส้นบนบัสทำหน้าที่ดังนี้:Slave Select (SS): สัญญาณที่แจ้งให้ slave ทราบว่า master กำลังทำงานร่วมกับมันSerial Clock (SCLK): สัญญาณนาฬิกาที่จัดลำดับการถ่ายโอนข้อมูลบนบัสทีละบิตในแต่ละรอบMaster Input Slave Output (MISO): สายที่ใช้ถ่ายโอนข้อมูลจาก slave ไปยัง masterMaster Output Slave Input (MOSI): สายที่ใช้ถ่ายโอนข้อมูลจาก master ไปยัง slave โดย SPI นั้นสามารถถ่ายโอนข้อมูลได้ทั้งสองทิศทางพร้อมกันบนสาย MISO และ MOSI ทำให้เป็นมาตรฐานการสื่อสารแบบฟูล-ดูเพล็กซ์ (full-duplex) หนึ่ง master สามารถควบคุมหลาย slave ได้ แต่ต้องใช้พิน SS ตัวอย่างในรูปที่ 23



รูปที่ 23 โปรโตคอล SPI

ที่มา: http://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/

3. **โปรโตคอลการสื่อสาร CAN** ระบบบัส CAN แบ่งออกเป็นสามประเภทตามความต้องการด้านอัตราการส่งสัญญาณและปริมาณข้อมูล:
    a. บัส CAN สำหรับการขับเคลื่อน (high-speed) ที่ 500 kbps ใช้สำหรับการสื่อสารที่เกือบเรียลไทม์
    b. บัส CAN สำหรับความสะดวก (low-speed) ที่ 100 kbps ใช้สำหรับการสื่อสารที่มีข้อกำหนดเวลาต่ำ
    c. บัส CAN สำหรับความบันเทิง (low-speed) ที่ 100 kbps ใช้สำหรับการสื่อสารที่มีข้อกำหนดเวลาต่ำ

นอกจากนั้น บัส CAN ใช้การเดินสายแบบคู่เกลียวและการถ่ายโอนข้อมูลแบบต่างกันเพื่อให้ความปลอดภัยสูงสุดในการถ่ายโอนข้อมูล โดยใช้ชื่อสายว่า CAN high และ CAN low โปรโตคอล CAN ใช้สำหรับการเชื่อมต่อส่วนประกอบในรถยนต์และยังมีการใช้งานในเครื่องบินสำหรับการวิเคราะห์ในระหว่างการบินและการเชื่อมต่อส่วนประกอบต่างๆ เช่น ระบบเชื้อเพลิงและปั๊ม CANopen เป็นที่นิยมในแอปพลิเคชันควบคุมฝังตัวในอุตสาหกรรม

บทที่ 3 หลักการและคุณลักษณะการสื่อสาร แบบอนุกรม (Serial Communication)

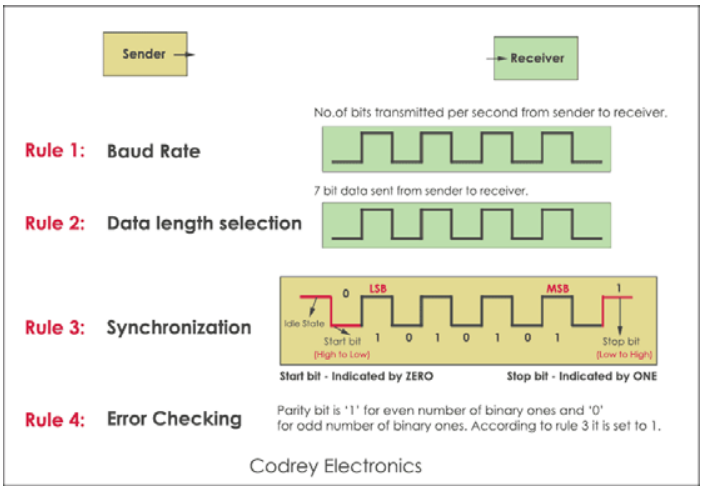
อย่างที่ได้แนะนำไปในบทที่ 2 โดย Serial Communication นั้นเป็นวิธีการสื่อสารที่ใช้สายการส่งข้อมูลเพียงหนึ่งหรือสองเส้นในการส่งและรับข้อมูล โดยข้อมูลจะถูกส่งและรับอย่างต่อเนื่องทีละบิต ข้อดีของการสื่อสารแบบอนุกรมคือสามารถเชื่อมต่อได้โดยใช้สายสัญญาณน้อย ทำให้ช่วยลดค่าใช้จ่ายในการใช้วัสดุสายไฟและอุปกรณ์ในการส่งสัญญาณ และสำหรับการสื่อสารในรูปแบบนี้นั้น จะมีกฎของการสื่อสารอยู่ด้วย

3.1 การทำงานการส่งข้อมูลแบบอนุกรม (Serial Communication Work)

ปัจจุบันซีพียูที่มีความก้าวหน้า เช่น ไมโครคอนโทรลเลอร์และไมโครโปรเซสเซอร์ ใช้การสื่อสารแบบอนุกรมเพื่อสื่อสารกับโลกภายนอกและอุปกรณ์ต่อพ่วงบนชิป สำหรับการทำความเข้าใจให้ดียิ่งขึ้น ลองพิจารณาตัวอย่างง่ายๆ เช่น หากคุณต้องการส่งไฟล์จากแล็ปท็อปของคุณไปยังสมาร์ทโฟน คุณจะส่งอย่างไร? อาจจะใช้โปรโตคอล Bluetooth หรือ WiFi ใช่ไหม ดังนั้น นี่คือขั้นตอนในการตั้งค่าการสื่อสารแบบอนุกรม 1. การเพิ่มการเชื่อมต่อ โดยในขั้นตอนแรก อุปกรณ์อิเล็กทรอนิกส์ (ตัวอย่างเช่น Tablet) จะค้นหาอุปกรณ์ที่อยู่ใกล้เคียงในระยะ 100 เมตร และแสดงรายการอุปกรณ์ที่พบ กระบวนการนี้มักเรียกว่าการค้นหา (roaming) 2. การเลือกอุปกรณ์ที่ต้องการสื่อสาร ทำเพื่อเชื่อมต่อกับอุปกรณ์อื่น (เช่น Smart phone) โดยต้องทำการจับคู่ (pairing) ซึ่งการตั้งค่าพื้นฐานจะถูกตั้งค่าไว้ในซอฟต์แวร์แล้ว ดังนั้นจึงไม่จำเป็นต้องตั้งค่าอัตราบอด (baud rate) ด้วยตนเอง นอกจากนี้ยังมีอีกสี่กฎที่ไม่คุ้นเคย ได้แก่ อัตราบอด, การเลือกบิตข้อมูล (framing), บิตเริ่มต้นและหยุด, และพาริตี้ (parity)

3.2 กฎเกณฑ์ของการส่งข้อมูลแบบอนุกรม (Rules of Serial Communication)

การส่งข้อมูลแบบ Serial นั้นจะมีกฎอยู่ 4 ข้อ ดังรูปที่ 24



รูปที่ 24 การส่งข้อมูลแบบอนุกรม

ที่มา: http://www.codrey.com/embedded-systems/serial-communication-basic/

3.1.1 อัตราบอด หรืออัตราความเร็วในการส่งข้อมูล (Baud rate)

อัตราบอด (Baud rate) คือ ความเร็วในการส่งข้อมูลจากตัวส่งไปยังตัวรับในรูปของบิตต่อวินาที อัตราบอดที่ใช้กันทั่วไปมีค่าเช่น 1200, 2400, 4800, 9600, และ 57600 นอกจากนั้นแล้วอัตราบอด (Baud rate) ยังเป็นตัว

ใช้กำหนดความต้องการแบนด์วิธในการส่งสัญญาณ และช่วยคำนวณอัตราบิต (Bit rate) ของช่องทางการสื่อสาร รวมถึงระบุความเร็วในการส่งข้อมูลผ่านสายอนุกรมหรืออินเตอร์เฟซอนุกรมได้ โดยมีวิธีการหาอัตราบอด ดังนี้

$$Baud\ rate = \frac{number\ of\ signal\ elements}{total\ time\ (in\ second)}$$

รูปแบบของอัตราความเร็วในการส่งข้อมูล ที่เรียกว่า อัตราบิต (Bit rate) ซึ่งเป็น จำนวนบิตไบนารี (1 หรือ 0) ที่ถูกส่งในแต่ละวินาที โดยมีวิธีการหาอัตราบิต ดังนี้

$$Bit\ rate = \frac{number\ of\ bit\ tranmitted}{total\ time\ (in\ second)}$$

นอกจากนั้นแล้ว อัตราบิต (Bit rate) ยังสามารถนิยามได้ตามอัตราบอด (Baud rate) ด้วยเช่นกัน ดังนี้

$$Bit\ rate = Baud\ rate\ \times bits\ per\ signal\ or\ symbol$$

3.2.1 การเลือกบิตข้อมูล (Framing Data)

การจัดกรอบข้อมูล (Framing) คือ การกำหนดจำนวนบิตที่ต้องการส่งจากอุปกรณ์โฮสต์ (เช่น แล็ปท็อป) ไปยังอุปกรณ์รับ (เช่น มือถือ) โดยทั่วไปมักใช้ 8 บิตในหลายอุปกรณ์ หลังจากเลือกจำนวนบิต (เช่น 8 บิต) แล้ว ต้องตกลงเรื่อง endianness ซึ่งเป็นลำดับการจัดเรียงข้อมูลระหว่างผู้ส่งและผู้รับเพื่อให้การถอดรหัสข้อมูลเป็นไปอย่างถูกต้อง

3.2.2 การซิงโครไนซ์ หรือ การกำหนดบิตเริ่มต้นและหยุด (Synchronization)

ในกระบวนการนั้น ตัวส่ง (Transmitter) จะเพิ่มบิตการซิงโครไนซ์ (1 บิตเริ่มต้นและ 1 หรือ 2 บิตหยุด) ลงในกรอบข้อมูลต้นฉบับ บิตการซิงโครไนซ์ช่วยให้ตัวรับ (Receiver) สามารถระบุจุดเริ่มต้นและจุดสิ้นสุดของการส่งข้อมูลได้ กระบวนการนี้เรียกว่าการส่งข้อมูลแบบไม่ต้องการการซิงโครไนซ์ (Asynchronous data transfer)

3.2.3 การควบคุมข้อผิดพลาด หรือ การตรวจสอบ parity (Error Control)

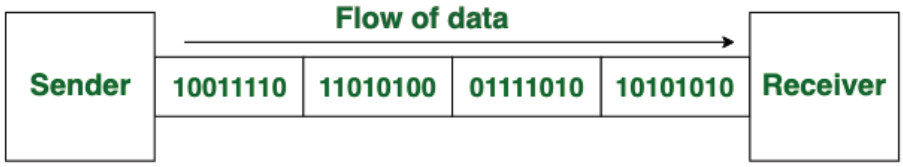
การเสียหายของข้อมูลอาจเกิดขึ้นเนื่องจากสัญญาณรบกวนจากภายนอกที่ปลายทางรับ วิธีเดียวที่จะแก้ไขเพื่อให้ได้ผลลัพธ์ที่เสถียรคือการตรวจสอบพาริตี้ (Parity) หากข้อมูลไบนารีมีจำนวน 1's เป็นเลขคู่จะเรียกว่า "even parity" และบิตพาริตี้จะถูกตั้งค่าเป็น '1' หากข้อมูลไบนารีมีจำนวน 1's เป็นเลขคี่จะเรียกว่า "odd parity" และบิตพาริตี้จะถูกตั้งค่าเป็น '0'"

3.3 วิธีการส่งข้อมูลแบบอนุกรม (Serial Data Transmission Method)

การสื่อสารแบบอนุกรมส่งข้อมูลเพียงบิตเดียวในแต่ละครั้ง ดังนั้นจึงต้องการสาย I/O (input-output) น้อยกว่า ซึ่งทำให้ใช้พื้นที่น้อยลงและทนทานต่อการขัดข้องจากสัญญาณข้าม (cross-talk) ได้ดี ข้อดีหลักของการ

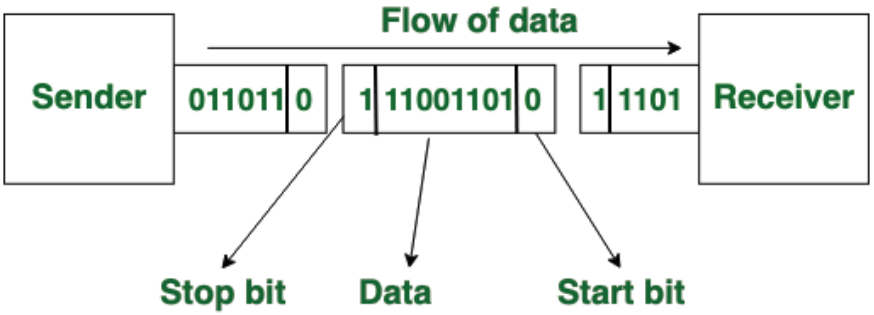
สื่อสารแบบอนุกรมคือ ค่าใช้จ่ายของระบบฝังตัวทั้งหมดจะลดลง และสามารถส่งข้อมูลไปยังระยะทางที่ไกลได้ การส่งข้อมูลแบบอนุกรมใช้ในอุปกรณ์ DCE (Data Communication Equipment) เช่น โมเด็ม และโดยทั่วไปแล้ว การสื่อสารแบบอนุกรม ใช้ 2 วิธีสำหรับการส่งและรับข้อมูล คือ

1. **Synchronous Transmission** คือการซิงโครไนซ์บิตและการซิงโครไนซ์บล็อก เมื่อสัญญาณข้อมูลถูกส่งผ่านไปตามสายสัญญาณเข้าสู่ DTE ปลายทาง สิ่งที่สำคัญที่สุดเลยคือ อุปกรณ์ปลายทางจะต้องสามารถส่งสัญญาณไบนารีหรือ บิต ซึ่งเป็นสัญญาณพื้นฐานที่สุดให้ได้อย่างถูกต้องตามจังหวะที่ส่งมา การที่อุปกรณ์ปลายทางสามารถรับ-ส่งอุปกรณ์ไบนารีได้ถูกต้องตามจังหวะนี้ เราเรียกว่า การซิงโครไนซ์บิต เมื่ออุปกรณ์ปลายทางสามารถรับ-ส่งบิตต่างๆได้อย่างถูกต้องแล้ว อุปกรณ์ปลายทางก็ยังจำเป็นต้องรู้ว่าสัญญาณที่รับมานั้น สำหรับแต่ละตัวมีการเริ่มต้นและสิ้นสุด เรียกว่า การซิงโครไนซ์บล็อกวิธิง่ายๆที่จะทำให้มีการซิงโครไนซ์บิต และการซิงโครไนซ์บล็อกนั้น ทำได้โดยการสงสัญญาณไทม์มิ่งไปยังสายอีกเส้นหนึ่งขนานไปกับสายที่ส่งสัญญาณข้อมูล แต่วิธีนี้จะไม่สามารถทำได้กรณีที่ติดต่อกัน เป็นระยะไกลๆ เนื่องจากมีค่าใช้จ่ายที่สูงมาก อีกทั้งวิธีนี้จะเป็นการส่งข้อมูลนี้เป็นแบบฟูลดูเพล็กซ์ (full-duplex) ซึ่งหมายความว่าการส่งและรับข้อมูลสามารถทำได้พร้อมกัน โดยการซิงโครไนซ์ระหว่างผู้ส่งและผู้รับเป็นสิ่งจำเป็น เพราะไม่มีช่วงเวลาห่างระหว่างข้อมูล อีกทั้งการส่งข้อมูลในลักษณะนี้มีประสิทธิภาพและความเชื่อถือได้สูงกว่าในการถ่ายโอนข้อมูลจำนวนมากหรือการสตรีมข้อมูล แสดงตัวอย่างการส่งข้อมูลในรูปที่ 25



รูปที่ 25 Synchronous Transmission
ที่มา: http://www.geekforgeeks.org/difference-between-synchronous-and-asynschronous-tranmission/

2. **Asynchronous Transmission** โดยทั่วไป Asynchronous เป็นคุณศัพท์อธิบายวัตถุหรือเหตุการณ์ที่ไม่มีพิกัดด้านเวลา ในเทคโนโลยีสารสนเทศ ศัพท์นี้มีการใช้หลายความหมาย ในสัญญาณการสื่อสารภายในเครือข่ายหรือระหว่างเครือข่าย สัญญาณ Asynchronous เป็นหนึ่งสัญญาณที่ส่งผ่านตามอัตรานาฬิกาต่างจากอีกสัญญาณหนึ่ง ในโปรแกรมคอมพิวเตอร์ปฏิบัติการ Asynchronous หมายถึงกระบวนการปฏิบัติงานอย่างอิสระของอีกกระบวนการขณะปฏิบัติการ Synchronous หมายถึงกระบวนการทำงานเฉพาะผลลัพธ์ของอีกกระบวนการที่เสร็จสิ้นหรือหยุดปฏิบัติการ กิจกรรมแบบแผนอาจจะใช้โปรโตคอล synchronous ที่จะส่งไฟล์จากจุดหนึ่งไปยังอีกจุดหนึ่ง แต่ละการส่งผ่านได้รับ การตอบสนองได้รับการส่งออกชี้ถึงความสำเร็จหรือต้องส่งใหม่ แต่การส่งผ่านสำเร็จของข้อมูลต้องการตอบสนองไปยังการส่งผ่านก่อนหน้านี้ก่อน อีกเริ่มต้นอีกกระบวนการ โดยรูปที่ 26 จะเป็นการแสดงตัวอย่างการส่งข้อมูลแบบ Asynchronous



รูปที่ 26 Asynchronous Transmission
ที่มา: http://www.geekforgeeks.org/difference-between-synchronous-and-asynschronous-tranmission/

สรุปแล้วการทำงานของ Synchronous และ Asynchronous วิธีส่งในทางปฏิบัตินั้น เมื่อแบ่งตามลักษณะการซิงโครไนซ์แล้วแบ่งได้เป็นการส่งแบบซิงโครนัส และการส่งแบบอะซิงโครนัส ซึ่งทั้งสองแบบนี้จะใช้วิธีแยกสัญญาณไทม์มิ่งจากสัญญาณข้อมูลที่รับมาการส่งแบบอะซิงโครนัสซึ่งส่วนใหญ่จะใช้กับระบบที่มีการส่งข้อมูลอัตราต่ำนั้น DTE ทางด้านส่งเมื่อต้องการส่งสัญญาณรหัสออกไป ก็จะมีการจัดสัญญาณนั้นให้อยู่ในรูปอนุกรมแล้วเติม

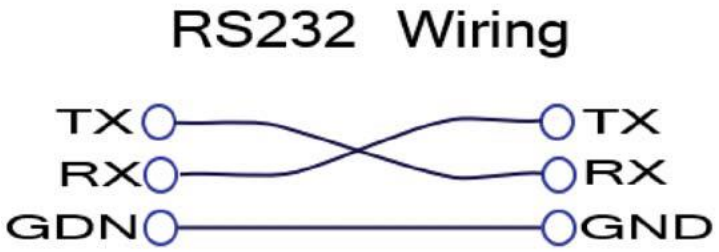
บิตเริ่มต้น (Start Bit) ไว้ที่ด้านหน้า และเติมบิตสิ้นสุด (Stop Bit) ไว้ที่ด้านหลัง และส่งออกไปตามจังหวะของสัญญาณนาฬิกาทางด้านส่ง ส่วย DTE ทางปลายทางนั้น เมื่อรับบิตเริ่มต้นได้ก็จะทำการรับสัญญาณข้อมูลที่ส่งตามมา โดยใช้จังหวะของสัญญาณนาฬิกาของสถานีตัวเอง และจะรับสัญญาณจนกว่าจะถึงบิตสิ้นสุดแล้วหยุดรับ ดังนั้นวิธีการนี้ bindtextdomain ก็จะทำการซิงโครไนซ์บิตและซิงโครไนซ์บล็อกพร้อมกันไป แต่วิธีนี้จะมีปัญหาเกิดขึ้นขึ้นได้ถ้าสัญญาณที่ส่งมามีความยาวมากขึ้น เพราะนั่นหมายถึงจังหวะสัญญาณนาฬิกาทางด้านการส่งและด้านรับจะมีโอกาศเบี่ยงเบนไปได้มากขึ้น เพราะฉะนั้นจึงมักใช้ส่งสัญญาณรหัสเป็นหน่วยสั้นๆ โดยตัวหนังสือสีฟ้าที่อยู่ในพารากราฟนี้จะเป็นข้อมูลต่างๆให้ท่านศึกษาเพิ่มเติมได้เผื่อท่านยังไม่เข้าใจและสงสัยในความหมายบางคำ

3.4 มาตรฐานการสื่อสารแบบอนุกรม (Serial Communication Standards)

มาตรฐานการสื่อสารแบบอนุกรม (Serial Communication Standards) เป็นข้อกำหนดที่กำหนดวิธีการส่งและรับข้อมูลผ่านช่องทางการสื่อสารแบบอนุกรม โดยทั่วไปจะมีการกำหนดรายละเอียดเช่น อัตราบอด (Baud rate), ขนาดของบิตข้อมูล, พาริตี้, และบิตเริ่มต้นและหยุด ตัวอย่างของมาตรฐานการสื่อสารแบบอนุกรม ได้แก่

3.4.1 RS-232

บัสอนุกรม RS-232 ประกอบด้วยสายสื่อสารเพียงสองเส้น คือ หนึ่งเส้นสำหรับส่งข้อมูลและอีกเส้นสำหรับรับข้อมูล ดังนั้นอุปกรณ์ที่ใช้การสื่อสารแบบอนุกรมควรมีขาอนุกรมสองขา รับข้อมูล (RX) - ส่งข้อมูล (TX) เนื่องจากข้อมูลการสื่อสารอิงตามแรงดันไฟฟ้าที่อยู่บนสายไฟซึ่งสัมพันธ์กับระดับพื้นดิน การเชื่อมต่อกับพื้นดิน (ground) จะต้องมีระหว่างอุปกรณ์ที่สื่อสารกันด้วย ตัวอย่าง RS-232 แสดงในรูปที่ 27



รูปที่ 27 การต่อสาย RS-232

ที่มา: http://www.raveon.com/wp-content/upload/2019/01/AN236SerailComm.pdf

3.4.2 RS-422

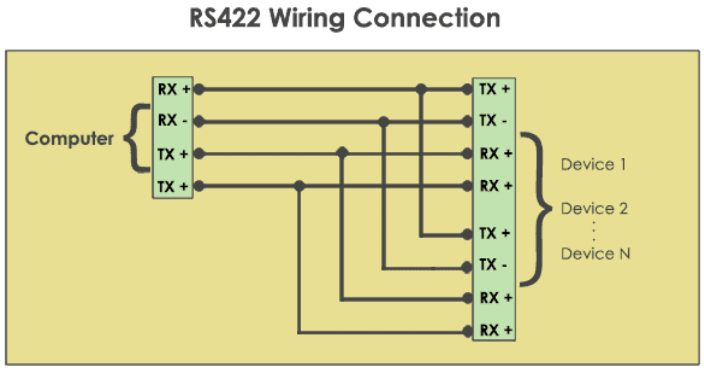
บัสอนุกรม RS-422 ได้รับการออกแบบให้ทนทานต่อสัญญาณรบกวนและสามารถใช้งานได้ดีในระยะสายเคเบิลที่ยาว มักใช้ระหว่างคู่ส่งข้อมูลและรับข้อมูลหนึ่งชุดกับอีกหนึ่งชุด ความต้านทานต่อสัญญาณรบกวนทำให้ระบบ RS-422 สามารถทำงานได้ดีในระยะทางที่ยาวกว่ามากเมื่อเปรียบเทียบกับ RS-232, US, และ Ethernet โดยแต่ละสัญญาณใช้สายสองเส้นในการส่งข้อมูล แรงดันไฟฟ้าเชิงต่าง (differential voltage) บนสาย A และ B แทนค่าดิจิทัล หาก B>A ค่าจะเป็น 1 หาก A>B ค่าจะเป็น 0 ตัวอย่าง RS-422 แสดงในรูปที่ 28

รูปที่ 28 การต่อสาย RS-422

ที่มา: http://www.raveon.com/wp-content/upload/2019/01/AN236SerailComm.pdf

3.4.3 RS-485

บัสอนุกรม RS-485 เป็นโปรโตคอลที่ได้รับความนิยมในอุตสาหกรรม แตกต่างจาก RS-422, คุณสามารถเชื่อมต่อไดรเวอร์สาย (line drivers) 32 ตัวและตัวรับข้อมูล (receivers) 32 ตัวในรูปแบบการเชื่อมต่อเชิงต่าง (differential configuration) โปรเซสเซอร์ (transmitter) ยังเรียกว่า ไดรเวอร์สาย (line driver) อย่างไรก็ตาม จะมีโปรเซสเซอร์ที่ทำงานอยู่เพียงตัวเดียวในแต่ละครั้ง แสดงในรูปที่ 29



รูปที่ 29 การต่อสาย RS-422

ที่มา: https://www.codrey.com/wp-content/uploads/2017/09/RS-422-Wiring-Connection-1.png

โดยสุดท้ายนี้จะเป็นการเปรียบเทียบข้อมูลของแต่ละชนิด ของทั้ง 3 มาตรฐานการสื่อสาร ได้แก่ RS-232, RS-422 และ RS-485 แสดงในรูปที่ 30

RS232, RS4422, and RS485 Comparision

Parameter	RS-232C	RS-422A	RS-485
Transmission mode	Simplex	Multi-point simplex	Multi-point multiplex
Max. connected devices	1 driver 1 receiver	1 driver 10 receivers	32 drivers 32 receivers
Max. transmission rate	20Kbps	10Mbps	10Mbps
Transmission method	Synchronous, Asynchronous	Synchronous, Asynchronous	Synchronous, Asynchronous
Max. cable length	15m	1200m	1200m
Operation mode	Single-ended (unbalanced type)	Differential (balanced type)	Differential (balanced type)
Features	Short distance Full-duplex 1:1 connection	Long distance Full-duplex, half-duplex 1:N connection	Long distance Full-duplex, half-duplex N:N connection

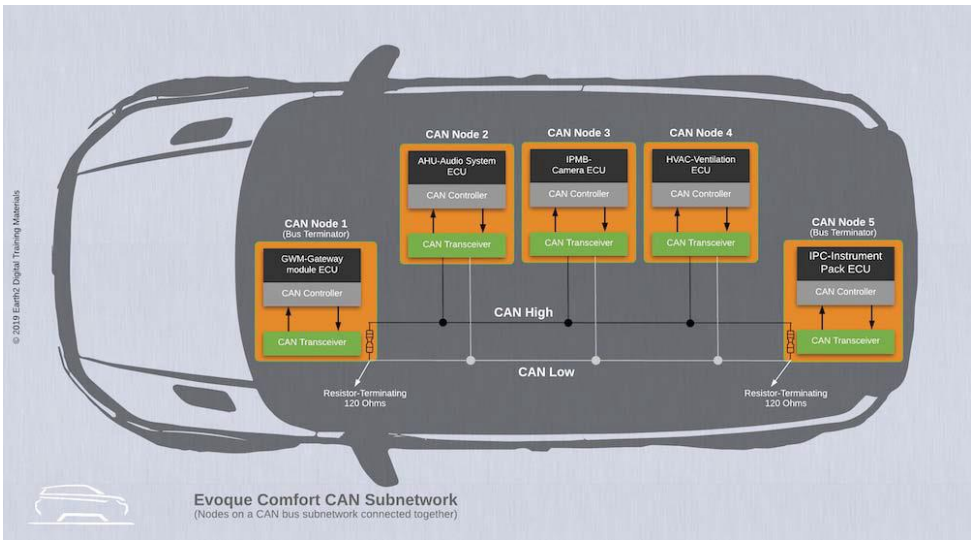
รูปที่ 30 การเปรียบเทียบ RS-232, RS-422 และ RS-485

3.5 การใช้งานการสื่อสารแบบอนุกรมในอุตสาหกรรมยานยนต์ (CAN และ LIN)

โปรโตคอลการสื่อสารแบบอนุกรมเปรียบเสมือนกระดูกสันหลังของการแลกเปลี่ยนข้อมูลในระบบอิเล็กทรอนิกส์หลายประเภท โดยให้วิธีการสำหรับอุปกรณ์ในการสื่อสารกันผ่านบัสที่ใช้ร่วมกัน เมื่ออุปกรณ์บนบัสปฏิบัติตามกฎชุดหนึ่งที่กำหนดโดยโปรโตคอล โปรโตคอลการสื่อสารแบบอนุกรมบนบัสจะวางรากฐานสำหรับโปรโตคอลการใช้งานในอุตสาหกรรมยานยนต์ ดังนี้

3.5.1 CAN (Controller Area Network)

CAN ย่อมาจาก Controller Area Network และเป็นโปรโตคอลการสื่อสารที่ใช้โดยอุปกรณ์อิเล็กทรอนิกส์หลายชนิด โดย CAN มักใช้ในการสื่อสารระหว่างอุปกรณ์ในรถยนต์ ตัวอย่างเช่น ระบบการจัดการเครื่องยนต์, ระบบกันสะเทือนแบบแอคทีฟ, ระบบ ABS, การควบคุมการเปลี่ยนเกียร์, การควบคุมไฟ, ระบบปรับอากาศ, ถุงลมนิรภัย, ระบบล็อกกลาง และระบบอื่นๆ ที่พบในยานยนต์ แสดงตัวอย่างการใช้งาน CAN ในรูปที่ 31

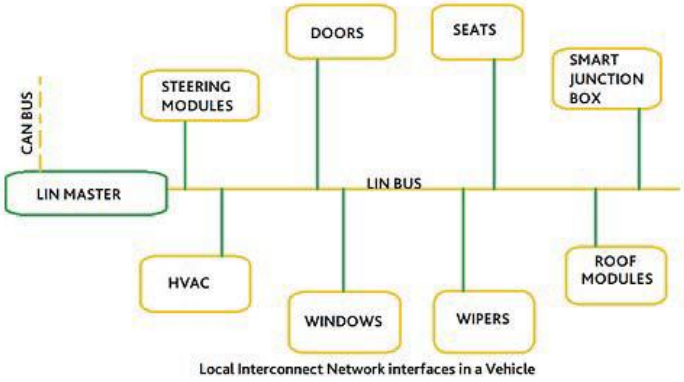


รูปที่ 31 CAN บนยานยนต์

ที่มา: http://www.earth2.digital/blog/what-is-vehicle-can-bus-ecu-evogue-adam-ali.html

3.5.2 LIN (Local Interconnect Network)

LIN ย่อมาจาก Local Interconnect Network และเป็นโปรโตคอลการสื่อสารอิเล็กทรอนิกส์ที่ใช้ในรถยนต์คล้ายกับ CAN แต่ความต้องการโปรโตคอล LIN เกิดขึ้นเนื่องจากบัสที่ใช้โปรโตคอล CAN มีค่าใช้จ่ายสูงเมื่อใช้กับอุปกรณ์ทั้งหมดในยานยนต์ที่ต้องการสื่อสารผ่านบัส แสดงตัวอย่างการใช้งาน LIN ในรูปที่ 32



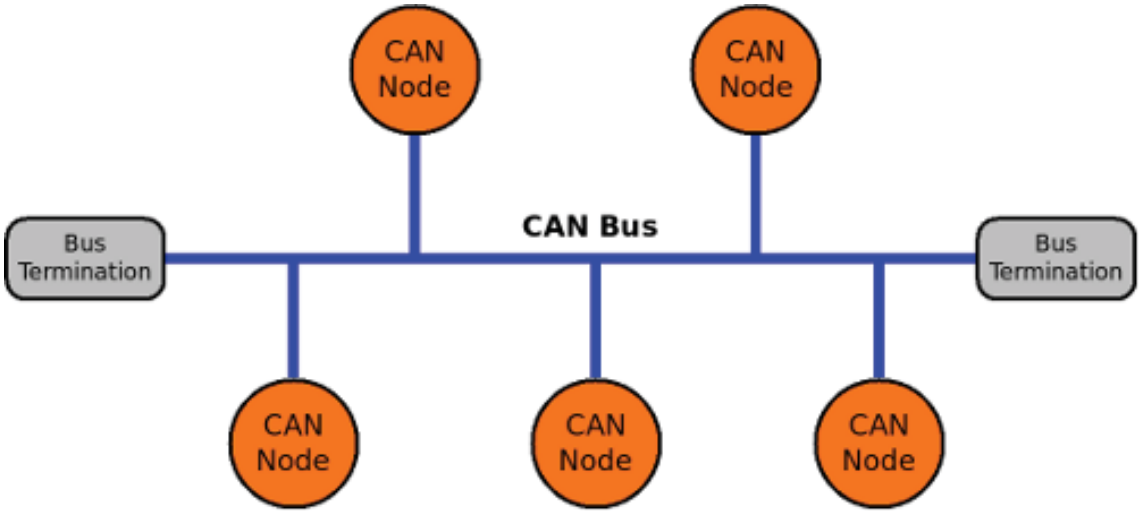
รูปที่ 32 การเชื่อต่อ LIN

ที่มา: http://www.influxbigdata.in/post/lin-protocol-a-simple-explanation

บทที่ 4 โปรโตคอลการสื่อสารในรถยนต์ กรณีศึกษา CAN และ LIN

4.1 Controller Area Network (CAN)

Controller Area Network หรือ CAN bus เป็นมาตรฐานการสื่อสารผ่านสายที่ออกแบบมาให้อุปกรณ์ใช้สื่อสารกับอุปกรณ์ โดยเริ่มต้นใช้อยู่ในยานยนต์เป็นหลัก ในยานยนต์แต่ละระบบจะมีส่วนควบคุม (ไมโครคอนโทรลเลอร์ หรือ กล่อง ECU) เป็นของตัวเอง เช่น เครื่องเล่นเพลง ระบบที่ปัดน้ำฝน ระบบถุงลมนิรภัย ระบบเครื่องยนต์ และระบบอื่น ๆ กว่า 70 ระบบ แต่ละระบบล้วนมีส่วนควบคุม (กล่อง ECU) เป็นของตัวเอง เพื่อให้ยานยนต์สามารถทำงานได้ ทุกระบบจึงจำเป็นจะต้องสื่อสารกัน และการสื่อสารกันก็ทำผ่าน CAN bus ได้ โดย CAN เป็นบัสการสื่อสารข้อมูลแบบอนุกรมที่มีความเสถียรสูง เหมาะสำหรับการใช้งานในเวลาจริง บัสนี้สามารถทำงานที่อัตราการส่งข้อมูลสูงสุดถึง 1 Mbps และมีความสามารถในการตรวจจับและแก้ไขข้อผิดพลาดที่ยอดเยี่ยม CAN ถูกพัฒนาโดย Bosch โดยมีการใช้งานหลักในอุตสาหกรรมยานยนต์ แต่ปัจจุบันยังใช้ในหลายแอปพลิเคชันในระบบอัตโนมัติและการควบคุมอุตสาหกรรมอีกด้วย โดย CAN เป็นโปรโตคอลแบบหลายมาสเตอร์ (multi-master) ที่อิงตามข้อความ (message-based) หมายความว่าอุปกรณ์ CAN ทุกตัวสามารถส่งข้อมูลได้ และหลายอุปกรณ์ CAN สามารถขอใช้บัสพร้อมกันได้ แสดงตัวอย่างการเชื่อมต่อในรูปแบบ CAN ดังรูปที่ 33



รูปที่ 33 Controller Area Network

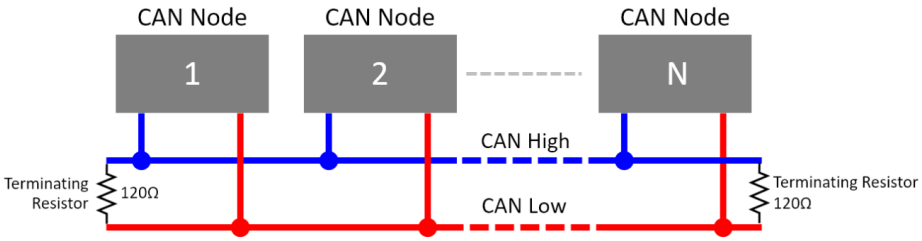
ที่มา: http://store.clipkin.com/article/can-bus-protocol-10-minute-lesson

CAN bus มีข้อได้เปรียบ และแตกต่างกับการสื่อสารผ่านสายแบบอื่น ๆ ดังนี้

1. **อุปกรณ์บนบัสคือตัวแม่ (Multi-master)** กล่าวคือ ปกติการสื่อสารผ่านสายที่เชื่อมต่ออุปกรณ์หลาย ๆ ตัวเข้าด้วยกันบนบัสเดียวกัน เช่น Modbus RTU , I2C , SPI จะต้องมีตัวแม่ (Master) เพียงตัวเดียว และตัวที่เหลือเป็นตัวลูก (Slave) โดยตัวลูกจะส่งข้อมูลได้ก็ต่อเมื่อตัวแม่ร้องขอเท่านั้น สำหรับ CAN bus อุปกรณ์ทุกตัวคือตัวแม่ การส่งข้อมูลจะส่งให้ใครเมื่อไรก็ได้ ไม่มีใครเป็นตัวควบคุม แก้ปัญหาหากตัวแม่พังเสียหาย ระบบจะไม่สามารถใช้งานได้ทั้งหมด (บน CAN bus หากมีอุปกรณ์ใดเสียหาย อุปกรณ์นั้น ๆ จะแยกตัวเองออกจากอุปกรณ์อื่น ๆทำให้ยังสื่อสารกันได้ปกติ)
2. **อุปกรณ์ได้รับข้อมูล แต่เลือกรับข้อมูลได้ (Multi-cast)** การส่งข้อมูลบน CAN bus คือการส่งที่ทุกอุปกรณ์ได้รับข้อมูลทั้งหมด (broadcast) ทั้งนี้หากอุปกรณ์ใดต้องการเลือกรับเฉพาะบางข้อมูล (Multi-cast) ก็สามารถทำได้เช่นกัน
3. **การตรวจจับความผิดพลาดและแจ้งเตือน (Error Detection and Signalling)** อุปกรณ์บน CAN bus จะมีการตรวจสอบข้อมูลที่วิ่งอยู่ในบัสเสนอ หากมีอุปกรณ์ใดตรวจพบความผิดพลาดของการส่งข้อมูล อุปกรณ์นั้นจะส่งข้อมูลแจ้งเตือนทันที
4. **การจัดลำดับความสำคัญของข้อมูล (Message Priorities)** หากเกิดเหตุการณ์ที่มีอุปกรณ์ใด ๆ ส่งข้อมูลพร้อมกันขึ้น ข้อมูลที่มีความสำคัญมากกว่าจะได้รับสิทธิ์ในการส่งก่อน ส่วนข้อมูลที่มีความสำคัญน้อยกว่าจะได้โอกาศส่งใหม่ในภายหลัง

4.1.1 CAN ในฐานะ Physical Layer

Physical Layer หรือ ชั้นกายภาพ เป็นข้อกำหนดการเชื่อมต่อสาย CAN bus ใช้สายในการเชื่อมต่อระหว่างส่วนควบคุม (ECU หรือไมโครคอนโทรลเลอร์หรือ CAN Device) ด้วยสาย 2 เส้น เชื่อมต่ออุปกรณ์ที่ต้องการสื่อสารทั้งหมดเข้าด้วยกัน ดังรูปที่ 34 ประกอบด้วยสาย CAN High (CANH) และ CAN Low (CANL) ที่ปลายสายทั้ง 2 ด้าน ต่อตัวต้านทาน 120Ω (เรียกว่า Terminating Resistor)



รูปที่ 34 การเชื่อมต่อระหว่างส่วนควบคุมด้วย CAN bus

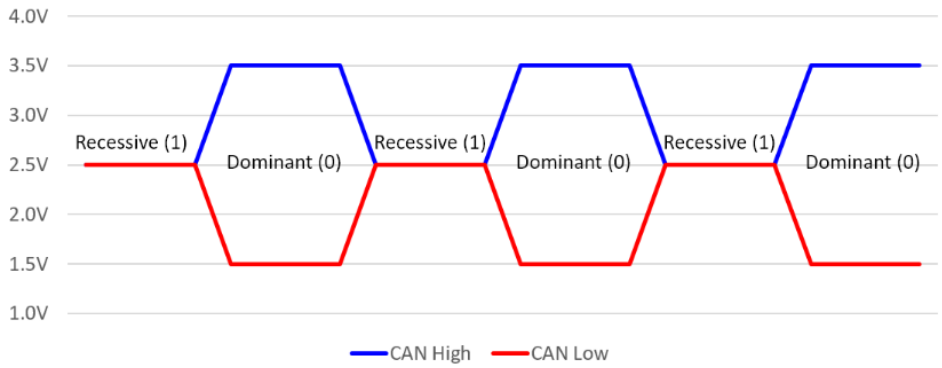
ที่มา: https://www.artronshop.co.th/article/101/can-bus

โดยสาย CANH และ CANL ทำงานแบบ differential wire คือใช้ความแตกต่างของแรงดันไฟฟ้าระหว่างสาย 2 เส้นในการรับ-ส่งข้อมูล โดยมีวัตถุประสงค์เพื่อลดสัญญาณรบกวน สัญญาณภายในสายประกอบด้วย 2 สถานะ คือ

1 **สถานะ Dominant** เกิดขึ้นเมื่อแรงดันของสาย CANH มากกว่าสาย CANL ซึ่งแปลเป็นสถานะส่งลอจิก 0

2 **สถานะ Recessive** เกิดขึ้นเมื่อแรงดันของสายเส้น CANH น้อยกว่าหรือเท่ากับ CANL แปลเป็นสถานะส่งลอจิก 1

และจากรูปที่ 35 จะเห็นว่า ในสถานะ Dominant (ส่งลอจิก 0) แรงดันของสาย CANH มีค่าประมาณ 3.5V ส่วน CANL มีค่าประมาณ 1.5V ในสถานะ Recessive (ส่งลอจิก 1) แรงดันของสาย CANH และ CANL มีค่าเท่ากันคือ 2.5V สาย CANH และ CANL มีแรงดันแตกต่างกัน 0V

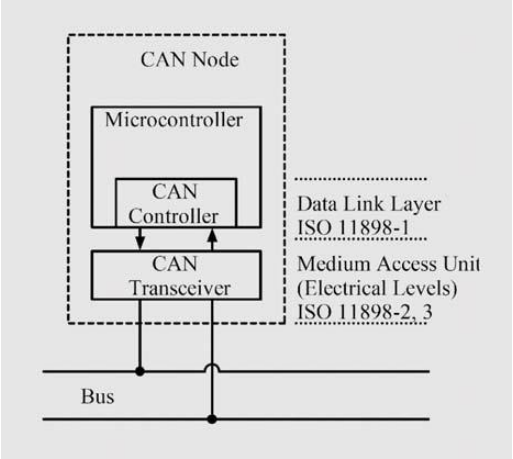


รูปที่ 35 แรงดันไฟฟ้าในสาย CAN High (CANH) และ CAN Low (CAN

ที่มา: https://www.artronshop.co.th/article/101/can-bus

4.1.2 โหนด (Node)

โหนด (Node) หรืออุปกรณ์ CAN (CAN Device) หรือส่วนควบคุม (ECU) แสดงตัวอย่างในรูปที่ 36 โดยภายในโหนดประกอบด้วย 2 ส่วน คือ



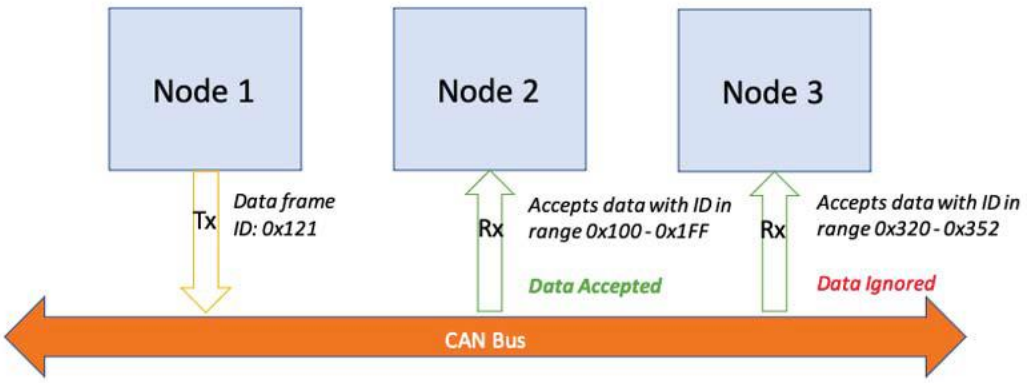
รูปที่ 36 ส่วนประกอบของ CAN Node

ที่มา: http://dewesoft.com/blog/what-is-can-bus

1. **CAN controller** เป็นวงจรไฟฟ้าที่ออกแบบมาสำหรับใช้ควบคุมการรับ-ส่งข้อมูลผ่าน CAN โดยเฉพาะ มักฝังมาภายในไมโครคอนโทรลเลอร์ที่มีประสิทธิภาพสูง หรือถูกออกแบบมาเพื่องานด้าน Automotive โดยเฉพาะ ไมโครคอนโทรลเลอร์ที่ฝัง CAN controller มาในตัว เช่น ESP32 ATSAME51 เป็นต้น ไมโครคอนโทรลเลอร์ที่ไม่มี CAN controller สามารถต่อไอซี (IC) CAN controller ภายนอกเพิ่มได้ เช่น MCP2515 เป็นต้น
2. **CAN Transceiver** ทำหน้าที่แปลงสัญญาณลอจิก 0 และ 1 แบบ TTL (ลอจิก 1 = 3.3V/5V, ลอจิก 0 = 0V) ให้เป็นสัญญาณเพื่อส่งออก CANH และ CANL โดย CAN Transceiver เป็นอุปกรณ์ที่ต้องต่อแยกออกมาจากตัวไมโครคอนโทรลเลอร์ ไอซี CAN Transceiver มีผลิตหลายบริษัท แต่ละบริษัทแต่ละรุ่นจะมีประสิทธิภาพด้านความเร็วในการรับ-ส่งข้อมูล (Data Rate) ที่แตกต่างกัน และใช้แรงดันไฟเลี้ยงต่างกัน (มีรุ่นใช้ไฟเลี้ยง 5V กับรุ่นใช้ไฟเลี้ยง 3.3V) ตัวอย่างไอซี CAN Transceiver ได้แก่ SN65HVD232DR TJA1050 BD41041FJ-CE2 โดยส่วนใหญ่ ไอซี CAN Transceiver ตัวถังจะเป็น SOIC-8 มีขาที่ตรงกันทุกเบอร์ ดังนั้นจึงใช้แทนกันได้ทั้งหมด

4.1.3 การรับ-ส่งข้อมูลผ่าน CAN (CAN Data Transmission)

กล่าวคือโปรโตคอล CAN เป็นแบบอิงข้อความ (message-based) หมายความว่าโหนดทั้งหมดบนบัสสามารถส่งและรับข้อความได้ และโหนดเหล่านั้นจะคอยรับฟังข้อความที่ถูกกระจายตลอดเวลา ดังรูปที่ 37



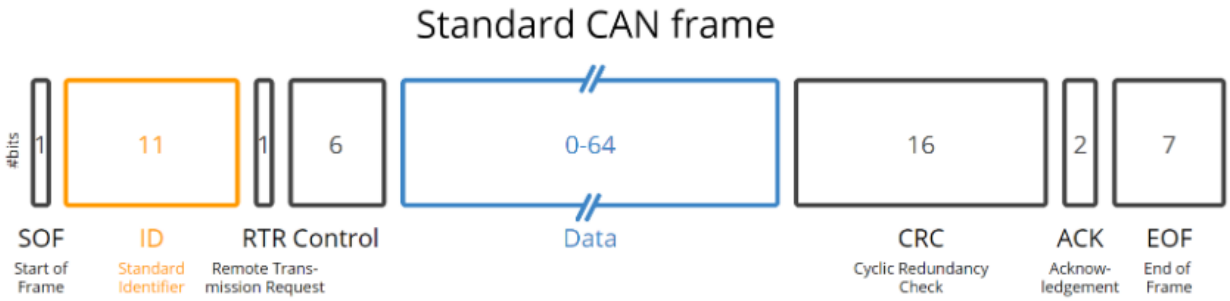
รูปที่ 37 การรับส่งข้อมูล CAN

ที่มา: https://community.nxp.com/t5/Blog/101-cController-Area-Network-CAN-standard/pa-p/1217054

และสำหรับการรับ-ส่งข้อมูลผ่าน CAN นั้นจะใช้สิ่งที่เรียกว่า CAN frame และ 1 CAN frame หมายถึง การส่งข้อมูล 1 ครั้ง โดย CAN frame แบ่งเป็น 4 ชนิดดังนี้ และมีรูปแบบโครงสร้างดังรูปที่ 38

1. Data frame – ใช้ส่งข้อมูลไปยังอุปกรณ์ CAN อื่น ๆ
2. Remote frame – ใช้ร้องขอข้อมูลจากอุปกรณ์ CAN อื่น ๆ

3. Error frame – ใช้แจ้งทุกอุปกรณ์ CAN ว่าพบความผิดพลาดขึ้นบนบัส
4. Overload frame



รูปที่ 38 ส่วนประกอบของ CAN frame
ที่มา: http://csselectronics.com

โครงสร้างของ CAN frame ประกอบไปด้วย

1. **SOF (Start of Frame) (1 บิต)** ส่ง Dominant (ส่งลอจิก 0) เพื่อบอกให้ทุกอุปกรณ์บนบัสรับรู้ว่ากำลังจะมีการส่งข้อมูล.
2. **ID (Identifier) (11 บิต สำหรับ Standard Frame)** เป็นหมายเลขเฉพาะของชุดข้อมูลนี้ โดยอาจกำหนดเป็นหมายเลขอุปกรณ์ หรือหมายเลขของข้อมูลที่ตั้งขึ้นมาเฉพาะก็ได้ โดยหมายเลขนี้จะเป็นตัวกำหนดความสำคัญของข้อมูลด้วย หากมีค่าน้อย หมายถึงมีความสำคัญมาก
3. **RTR (Remote Transmission Request) (1 บิต)** เป็นบิตที่กำหนดว่าเป็น Remote frame หรือไม่ ดังนี้

RTR = Dominant (ลอจิก 0) หมายถึง ข้อมูลชุดนี้เป็น Data frame

RTR = Recessive (ลอจิก 1) หมายถึง ข้อมูลชุดนี้เป็น Remote frame

Control (6 บิต) ประกอบไปด้วย

IDE (Identifier extension bit) (1 บิต) ใช้ตรวจจับว่าเป็น Standard Frame หรือ Extended Frame

Reserved (1 บิต) สงวนบิตนี้ไว้สำหรับฟีเจอร์ในอนาคต มีค่าเป็น Dominant (ลอจิก 0)

DLC (Data Length Code) (4 บิต) แบ่งเป็น 2 กรณีดังนี้

ถ้า **RTR** = Dominant (ลอจิก 0, Data frame) ใช้บอกความยาวของข้อมูลที่ส่ง ถ้า **RTR** = Recessive (ลอจิก 1, Remote frame) ใช้บอกความยาวของข้อมูลที่ร้องขอ

Data (0 ถึง 64 บิต หรือ 0 ถึง 8 ไบต์) คือข้อมูลที่ต้องการส่ง

CRC (16 บิต) – แบ่งดังนี้

CRC (15 บิต)

CRC delimiter (1 บิต) – มีค่าเป็น Recessive (ลอจิก 1) เสมอ

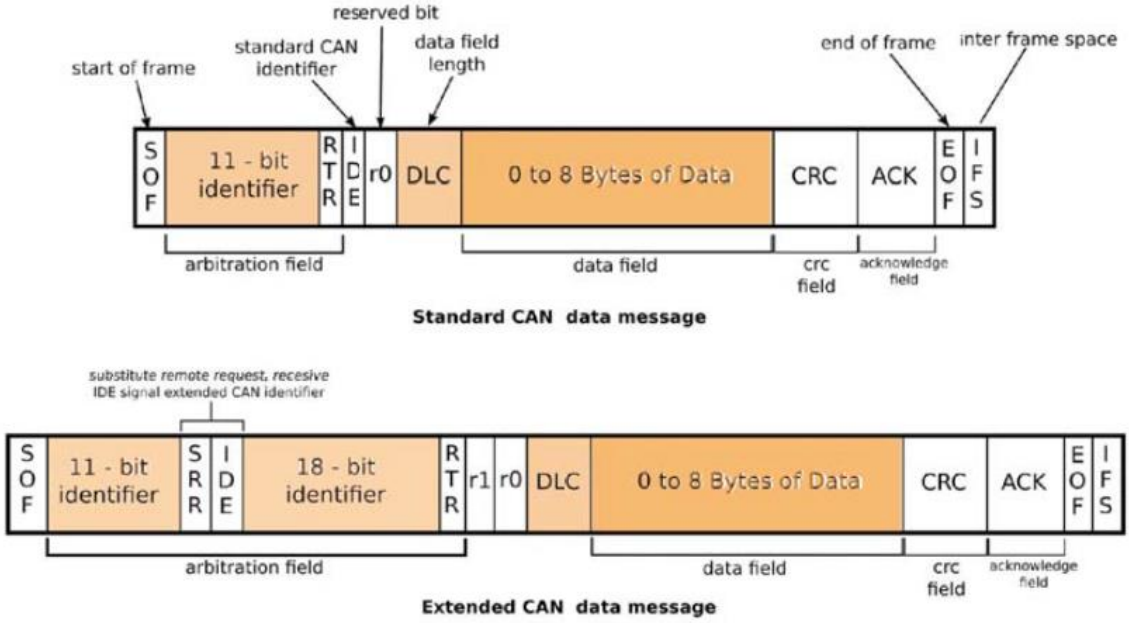
ACK (2 บิต) – แบ่งดังนี้

ACK slot (1 บิต) – ฝั่งส่งจะปล่อยให้บัสเป็นสถานะ Recessive (ลอจิก 1) หากมีอุปกรณ์ใด ๆ ในบัสได้รับและยืนยันว่าข้อมูลที่ส่งถูกต้อง บิตนี้จะถูกดึงเป็น Dominant (ลอจิก 0)

ACK delimiter (1 บิต) – มีค่าเป็น Recessive (ลอจิก 1) เสมอ

EOF (7 บิต) ใช้ส่งเพื่อบอกสิ้นสุด CAN frame

ในด้านการใช้งานจริงมีเพียง Data frame และ Remote frame เท่านั้นที่ผู้ใช้สามารถเขียนโปรแกรมสั่งงานได้ ส่วน Error frame และ Overload frame ตัว CAN controller จะจัดการให้อัตโนมัติ โดยData frame และ Remote frame มี 2 รูปแบบ คือ Standard Frame และ Extended Frame โดย Standard Frame มีโครงสร้างดังรูปที่ 39 ซึ่ง Extended Frame แตกต่างจาก Standard Frame ตรงที่ Arbitration field มีความยาว 29 บิต และมีโครงสร้างอื่น ๆ ที่แตกต่างกันเล็กน้อย



รูปที่ 39 Standard และ Extended ของ CAN
ที่มา: http://dewesoft.com/blog/what-is-can-bus

4.1.4 ความเร็วในการรับ-ส่งข้อมูล

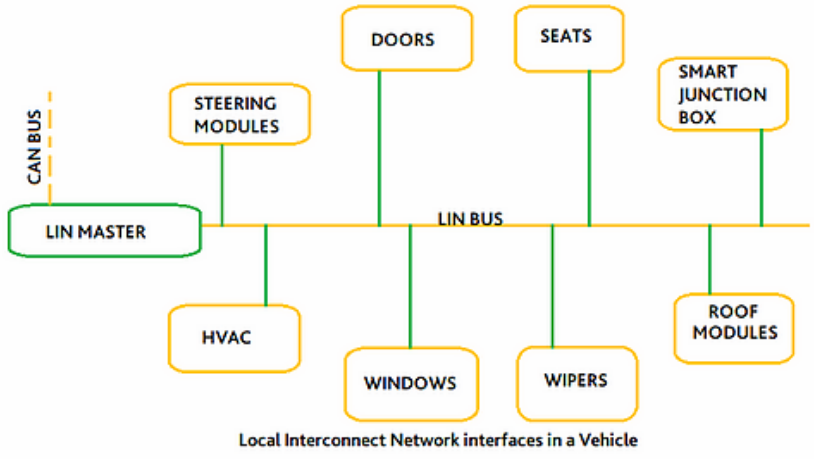
ความเร็วในการรับ-ส่งข้อมูลจะถูกกำหนดตาม Data Rate หรือ Baud rate ซึ่งสามารถตั้งค่าได้ตามตัวเลือกต่างๆ เช่น 12.5 kbit/s, 16 kbit/s, 20 kbit/s, 25 kbit/s, 50 kbit/s, 100 kbit/s, 125 kbit/s (ซึ่งมักใช้บ่อย), 250 kbit/s, 500 kbit/s, 800 kbit/s, และ 1 Mbit/s การกำหนด Data Rate หรือ Baud rate ของอุปกรณ์ทุกตัวบนบัส CAN ต้องเป็นค่าที่เท่ากันเพื่อให้การรับ-ส่งข้อมูลถูกต้อง นอกจากนี้แล้วความยาวของสายเคเบิลยังส่งผลกับความเร็วในการส่งข้อมูลอีกด้วย

4.2 Local Interconnect Network (LIN)

Local Interconnect Network (LIN) ถูกพัฒนาเพื่อสร้างมาตรฐานสำหรับการสื่อสารแบบมัลติเพล็กซ์ที่มีต้นทุนต่ำและประสิทธิภาพต่ำในเครือข่ายยานยนต์ ถึงแม้ว่า Controller Area Network (CAN) จะตอบสนองความต้องการของเครือข่ายที่มีแบนด์วิธสูงและการจัดการข้อผิดพลาดที่ซับซ้อน แต่ต้นทุนฮาร์ดแวร์และซอฟต์แวร์ของการใช้ CAN ได้กลายเป็นอุปสรรคสำหรับอุปกรณ์ที่มีประสิทธิภาพต่ำ เช่น ควบคุมกระจกไฟฟ้าและที่นั่ง LIN ให้การสื่อสารที่คุ้มค่าในแอปพลิเคชันที่ไม่ต้องการแบนด์วิธและความยืดหยุ่นของ CAN

โดยสามารถใช้ LIN ที่มีต้นทุนต่ำโดยใช้ UART (Universal Asynchronous Receiver/Transmitter) ที่ฝังอยู่ในไมโครคอนโทรลเลอร์ 8 บิตราคาต่ำในยุคสมัยใหม่นี้ อีกทั้งเครือข่ายยานยนต์สมัยใหม่ใช้การรวมกันของ LIN สำหรับแอปพลิเคชันที่มีต้นทุนต่ำโดยเฉพาะในอิเล็กทรอนิกส์ของตัวถัง แต่ใช้ CAN สำหรับการสื่อสารพลังงานหลักและตัวถัง รวมถึงบัส FlexRay ที่กำลังเกิดใหม่สำหรับการสื่อสารข้อมูลที่ซิงโครไนซ์ความเร็วสูงในระบบขั้นสูง เช่น ระบบกันสะเทือนแบบแอคทีฟ

หลักของ LIN ใช้แนวทางมาสเตอร์ (Master) และสเลฟ (Slave) ซึ่งประกอบด้วย LIN master หนึ่งตัว และ LIN slave มากกว่าหนึ่งตัวขึ้นไป ดังรูปที่ 40



รูปที่ 40 LIN สำหรับยานยนต์

ที่มา: https://www.influxtechnology.com/post/lin-basics

4.2.1 รูปแบบกรอบข้อมูลของ LIN (LIN Frame Format)

LIN นั้นจะใช้การสำรวจเพื่อตรวจสอบ (polling) จากอุปกรณ์มาสเตอร์หนึ่งตัวและอุปกรณ์สเลฟหลายตัว กล่าวคือการสื่อสารจะถูกควบคุมโดยมาสเตอร์ซึ่งส่งหัวข้อข้อมูล (header) ดังรูปที่ 41 ที่ประกอบด้วย

1. **เบรก (break)** ในทุก LIN frame จะเริ่มต้นด้วยเบรก (break) ซึ่งประกอบด้วย 13 บิตลอจิก 0 (dominant) ตามด้วยตัวแบ่งเบรก (break delimiter) ขนาด 1 บิต (ลอจิก 1) นี่เป็นการแจ้งเตือนเริ่มต้นกรอบข้อมูลให้กับทุกโหนดบนบัส

2. **การซิงโครไนซ์ (sync)** เป็นฟิลด์ที่สองที่ส่งโดยงานมาสเตอร์ในหัวข้อข้อมูล การซิงค์ถูกกำหนดให้เป็นอักขระ โดยฟิลด์ซิงค์จะช่วยให้สเลฟอุปกรณ์ที่ทำการตรวจจับอัตราบอด (Buad rate) อัตโนมัติสามารถวัดระยะเวลาของอัตราบอดและปรับอัตราบอดภายในของตนให้ตรงกับบัส
3. **ไอดี (ID)** เป็นฟิลด์สุดท้ายที่ส่งโดยมาสเตอร์ในหัวข้อข้อมูล ฟิลด์นี้ให้การระบุสำหรับแต่ละข้อความบนเครือข่ายและกำหนดว่าโหนดไหนในเครือข่ายจะรับหรือตอบสนองต่อการส่งข้อมูลแต่ละครั้ง งานสเลฟทั้งหมดจะฟังฟิลด์ ID อย่างต่อเนื่อง เพื่อตรวจสอบพาริตี้ (Parities) และกำหนดว่าตนเป็นผู้เผยแพร่หรือผู้สมัครเข้าร่วม สำหรับหมายเลขการระบุนี้ LIN มีทั้งหมด 64 IDs โดย ID 0 ถึง 59 ใช้สำหรับกรอบข้อมูลที่ส่งสัญญาณ (data frames) ส่วน ID 60 และ 61 ใช้สำหรับข้อมูลการวินิจฉัย ID 62 สำรองไว้สำหรับการขยายที่ผู้ใช้กำหนด และ ID 63 สำรองไว้สำหรับการพัฒนาโปรโตคอลในอนาคต
4. **ข้อมูล (data payload)** ฟิลด์นี้จะถูกส่งโดยงานสเลฟในระหว่างการตอบสนอง ฟิลด์นี้ประกอบด้วยข้อมูลที่มีตั้งแต่หนึ่งถึงแปดไบต์ของข้อมูลภาระ (payload data bytes)
5. **เช็คซัม (checksum)** ฟิลด์นี้จะถูกส่งโดยสเลฟในระหว่างการตอบสนอง โดย LIN 0tกำหนดให้ใช้หนึ่งในสองอัลกอริธึมเช็คซัมเพื่อคำนวณค่าที่อยู่ในฟิลด์เช็คซัม 8 บิต อัลกอริธึมเช็คซัมแบบคลาสสิก (Classic checksum) จะคำนวณจากการรวมค่าของข้อมูลไบต์เท่านั้น ส่วนอัลกอริธึมเช็คซัมแบบพัฒนา (Enhanced checksum) จะคำนวณจากการรวมค่าของข้อมูลไบต์และไอดีที่ได้รับการปกป้อง

รูปที่ 41 ส่วนประกอบของ LIN frame

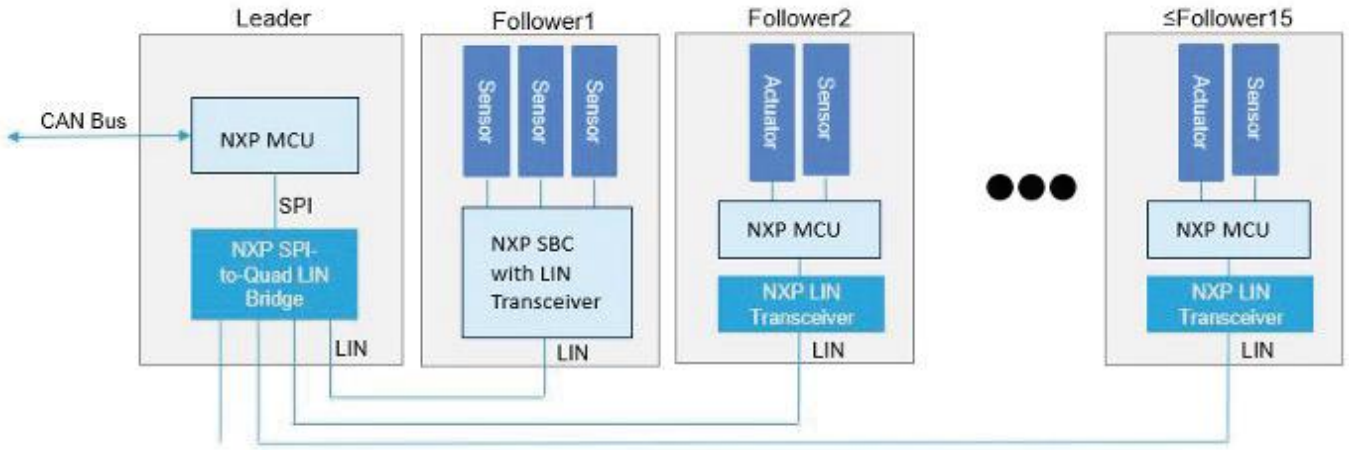
ที่มา: https://www.ni.com/en/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/

ในการทำงานนั้นมาสเตอร์จะตรวจสอบงานสเลฟในลูป โดยส่งข้อมูลที่ประกอบด้วยลำดับเบรกม, ซิงค์ และไอดี ส่วนสเลฟนั้นจะตรวจสอบไอดีและดำเนินการตามที่กำหนด หากสเลฟต้องการตอบสนองข้อมูลกลับ จะส่งข้อมูล 1 ถึง 8 ไบต์พร้อมเช็คซัม หากสเลฟตัวอื่นต้องการจะเข้าร่วม จะอ่านข้อมูลและเช็คซัมจากบัส และดำเนินการที่เหมาะสมต่อไป สำหรับการสื่อสารจากสเลฟไปยังมาสเตอร์ มาสเตอร์จะกระจายไอดีไปยังเครือข่าย และสเลฟตัวเดียวจะตอบกลับด้วยข้อมูล อีกทั้งในกรณีที่มาสเตอร์ต้องการส่งข้อมูลจะมีงานสเลฟภายในมาสเตอร์โหนดที่ตอบสนองเหมือนเป็นสเลฟอิสระ โดยการอัปเดตข้อมูลและเผยแพร่หัวข้อข้อมูลที่เหมาะสม

4.2.2 การจัดระเบียบและพฤติกรรมของ LIN (LIN Topology and Behavior)

อย่างที่ทราบกันดีว่า LIN นั้น ประกอบไปด้วยสเลฟ (Slave) หลายตัว จึงจำเป็นต้องมีการจัดการ โดย LIN มีหลักการดังนี้

1. **การเชื่อมต่อของบัส LIN** โดย LIN (Local Interconnect Network) ใช้เพื่อเชื่อมต่ออุปกรณ์มาสเตอร์ (โหนดหลัก) และอุปกรณ์สเลฟ (โหนดรอง) หลายตัวในเครือข่ายเดียวกันหรือที่เรียกว่า **คลัสเตอร์ LIN** ดังรูปที่ 42
2.



รูปที่ 42 LIN Cluster

ที่มา: http://community.nxp.com/t5/blog/101-Local-interconncet-network-LIN/

3. **การกำหนดพฤติกรรมของโหนด** พฤติกรรมของแต่ละโหนดในเครือข่ายจะถูกระบุใน ไฟล์ความสามารถของโหนด (Node Capability File) ไฟล์เหล่านี้จะถูกใช้เพื่อสร้าง ไฟล์คำอธิบาย LIN (LDF) ซึ่งบรรยายถึงพฤติกรรมของทั้งคลัสเตอร์
4. **การทำงานของไฟล์ LDF** ไฟล์ LDF จะถูกอ่านโดยเครื่องมือสร้างระบบ เพื่อสร้างพฤติกรรมที่ระบุในโหนดต่างๆ มาสเตอร์จะเริ่มส่งหัวข้อข้อมูล (header) บนบัส และสเลฟทั้งหมดในคลัสเตอร์จะตอบสนองตามที่ระบุใน LDF
5. **การกำหนดค่าและการจัดตาราง** ไฟล์ LDF ใช้ในการกำหนดค่าและจัดตารางการทำงานของคลัสเตอร์ LIN เช่น การกำหนดอัตราบอด (baud rate) และการจัดลำดับการส่งข้อมูล อุปกรณ์ NI LIN และ NI-CAN Frame API สำหรับ LIN ไม่สนับสนุนในการดาวน์โหลดพฤติกรรมการจัดตารางไปยังฮาร์ดแวร์โดยตรง
6. **การสนับสนุนจาก NI** อุปกรณ์ NI LIN มีคิวการตอบสนองสำหรับเก็บการตอบสนองของสเลฟ รองรับได้ถึง 64 การตอบสนอง (ตามจำนวนไอดีสูงสุด 64 ตัวที่ระบุสำหรับ LIN) การตอบสนองต้องเกิดขึ้นภายในเวลาที่กำหนดโดยข้อกำหนดของ LIN

7. **ความสามารถของ NI-CAN Frame API สำหรับ LIN** NI-CAN Frame API สำหรับ LIN ให้ฟังก์ชันพื้นฐานในการโต้ตอบกับ LIN ในระดับที่ต่ำ เนื่องจากไม่สนับสนุนการวินิจฉัย หรือการกำหนดค่า LIN และ LDFs โดยตรง แต่สามารถใช้ในการพัฒนาแอปพลิเคชันที่ซับซ้อนเกี่ยวกับเครือข่าย LIN ได้

โดยสรุป LIN ช่วยในการเชื่อมต่อและการสื่อสารระหว่างอุปกรณ์ในเครือข่าย โดยการจัดการพฤติกรรมผ่านไฟล์ LDF และเครื่องมือของ NI ช่วยให้การทำงานร่วมกับบัส LIN เป็นไปได้อย่างมีประสิทธิภาพ แม้จะมีข้อจำกัดในบางด้าน.

4.2.3 การตรวจจับข้อผิดพลาดและการควบคุมข้อผิดพลาดใน LIN (LIN Error Detection and Confinement)

ข้อกำหนดของ LIN ระบุว่าการตรวจจับข้อผิดพลาดควรถูกจัดการโดยสเลฟ และการตรวจสอบข้อผิดพลาดโดยมาสเตอร์นั้นไม่จำเป็น กล่าวคือข้อกำหนด LIN ไม่จำเป็นต้องจัดการกับข้อผิดพลาดหลายๆ รายการในเฟรมเดียวหรือการใช้เคาน์เตอร์ข้อผิดพลาด เมื่อตรวจพบข้อผิดพลาดแรกในเฟรม สเลฟจะยุติการประมวลผลของเฟรมจนกว่าจะตรวจพบลำดับเบรก และซิงค์ถัดไป (ในหัวข้อข้อมูลถัดไปที่ส่งโดยมาสเตอร์) หากการตั้งค่าบันทึกข้อผิดพลาดบัสถูกตั้งค่าเป็นจริง เฟรมที่เกิดข้อผิดพลาดจะถูกบันทึกลงในคิวการอ่าน หากการตั้งค่าบันทึกข้อผิดพลาดบัสถูกตั้งค่าเป็นเท็จ ข้อผิดพลาดจะถูกส่งคืนโดยฟังก์ชัน ncWriteNet หรือ ncWriteNetMult อีกทั้ง LIN ยังมีการรายงานข้อผิดพลาดไปยังเครือข่าย ข้อกำหนด LIN กำหนดบิตสถานะ Response_Error ซึ่งสเลฟต้องรายงานให้มาสเตอร์ในหนึ่งในเฟรมที่ส่ง บิตนี้จะถูกตั้งค่าเมื่อเฟรมที่ได้รับหรือส่งโดยโหนดสเลฟมีข้อผิดพลาดในฟิลด์การตอบสนอง บิตนี้จะถูกล้างหลังจากที่มันถูกส่งในหนึ่งในการตอบสนองที่เผยแพร่ของสเลฟ NI-CAN Frame API สำหรับ LIN ไม่รองรับบิตสถานะ Response_Error โดยตรง แต่จะใช้วิธีการที่ง่ายในการใช้งานฟังก์ชันนี้ที่ระดับแอปพลิเคชัน ขั้นตอนการทำงานคือการตั้งค่าบันทึกข้อผิดพลาดบัสเป็น 1 เพื่อเปิดใช้งานการบันทึกเฟรมข้อผิดพลาดบัสในคิวอ่าน จากนั้นแอปพลิเคชันสามารถตรวจสอบการอ่านเฟรมข้อผิดพลาดบัสโดยใช้รหัสข้อผิดพลาดที่ระบุข้อผิดพลาดในการตอบสนอง เมื่อตรงตามเงื่อนไขนี้ แอปพลิเคชันสามารถตั้งค่าบิตสถานะ Response_Error ในตัวแปรภายใน และใช้ประเภทเฟรมการตอบสนอง NI LIN เพื่อต่ออายุคิวการตอบสนองของสเลฟด้วยข้อมูลที่มีบิตสถานะ Response_Error และล้างบิตนี้ในตัวแปรภายใน

4.3 การเปรียบเทียบโปรโตคอลบัส CAN และ LIN (CAN And LIN Bus Protocols Comparison)

ในท้ายที่สุดจะเป็นการเปรียบเทียบการทำงานของ CAN และ LIN เพื่อช่วยให้ง่ายต่อการเลือกใช้ และเหมาะสมกับงาน โดยแสดงในรูปที่ 43 โดยการเลือกใช้ระหว่าง CAN และ LIN ขึ้นอยู่กับข้อกำหนดของระบบ เช่น ความเร็วในการสื่อสาร, ความต้องการในการตรวจจับข้อผิดพลาด, และงบประมาณในการพัฒนา กล่าวคือ CAN เหมาะสำหรับระบบที่ต้องการการสื่อสารที่รวดเร็วและมีความเชื่อถือได้สูง แต่มีค่าใช้จ่ายสูง ส่วน LIN เหมาะสำหรับระบบที่มีค่าใช้จ่ายต่ำและความเร็วในการสื่อสารไม่สูง โดยมีค่าใช้จ่ายต่ำกว่าและใช้งานง่าย

Parameters	CAN	LIN
Full form	Controller Area Network	Local Interconnect Network
Communication type	Broadcast type	Master-slave
Communication network	Multi master. peer to peer	Single master and multiple slaves
No. of lines required	2 wires	1 wire
Maximum communication speed	1 Mbps	20 kbps
Triggering technique	Event triggered	Time triggered
Electromagnetic interference (EMI)	Resilient to EMI as it uses differential signaling	Less immune to EMI compared to CAN
Applications	Safety critical- Airbag, ABS, Engine Management System	Non-safety critical - Entertainment system, wiper control, mirror control

รูปที่ 43 การเปรียบเทียบ CAN และ LIN